YOUR OFFICIAL RASPBERRY PI MAGAZINE

# The MagPi

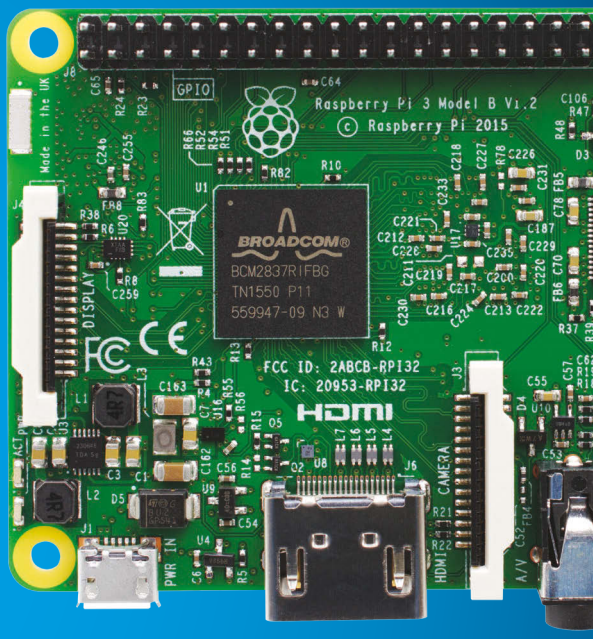
The official Raspberry Pi magazine

Issue 53 January 2017

raspberrypi.org/magpi

# BEGINNER'S GUIDE TO CODING

Discover the joy of programming with the biggest and best introductory guide for the Raspberry Pi




## USE THE PI FOR GOOD

BOINC turns an idling Pi into a force for science

## GOOGLE DEEPDREAMS

Create surreal pictures with your Raspberry Pi with Google's help

## DRESS TO IMPRESS

The social media-powered interactive dress

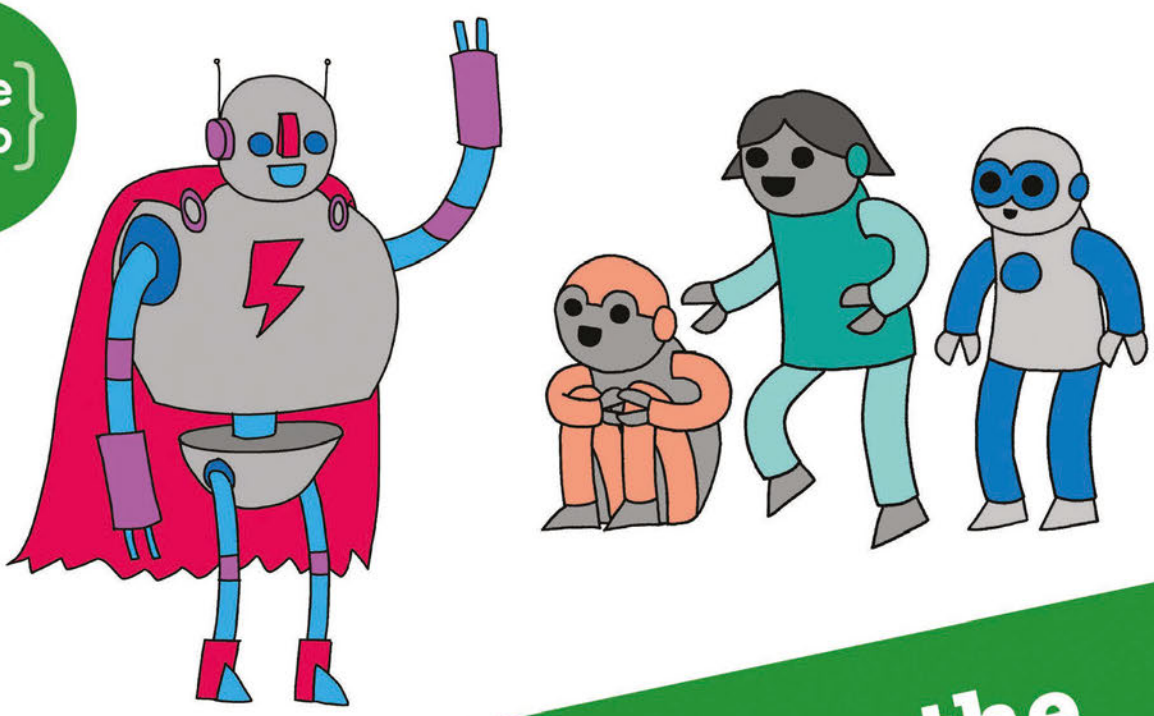
## RASPBERRY PI CAMERA MODULE 101

Take pictures and make videos in easy steps

Issue 53 • Jan 2017 • £5.99

9 772051 998001 01

THE ONLY PI MAGAZINE WRITTEN BY THE RASPBERRY PI COMMUNITY



# Can you help inspire the next generation of coders?

**Code Club** is a nationwide network of volunteer-led after school clubs for children aged 9-11.

We're always looking for people with coding skills to volunteer to run a club at their local primary school, library or community centre for an hour a week.

You can team up with colleagues, a teacher will be there to support you and we provide all the materials you'll need to help get children excited about digital making.

There are loads of ways to get involved!

So to find out more, join us at **www.codeclub.org.uk**

This advertising space was kindly donated to Code Club by MagPi Magazine

# WELCOME TO THE OFFICIAL MAGAZINE

Learning to code is one of the most profoundly life-changing things you can do. So says Lucy Hattersley in her beginner's guide to coding, starting on page 14.

She's right too; I distinctly remember the feeling of euphoria I got from finishing my first coding project from scratch. It must be like being a young Harry Potter learning of his life beyond the cupboard under the stairs. Realising that you have the power to conjure up something entirely new and unique from your fingertips is about as life-changing as it comes.

While we lack the witches and broomsticks of the wizarding world, we're increasingly surrounded by computers that do our bidding and much of our thinking for us. It's becoming urgently important that we understand their common tongue. While there's a lifetime of learning out there, the magazine you're holding can – believe it or not – equip you with the skills you need to write just about any game or application you can imagine. Yes, even that one!

**Russell Barnes**
**Managing Editor**

FREE DEBIAN + PIXEL LIVE BOOTING DVD
The MagPi
BEGINNER'S GUIDE TO CODING
Debian
PIXEL
The Raspberry Pi desktop
WANT TO KNOW HOW TO USE YOUR FREE DVD?
TURN TO PAGE 70
SUBSCRIBE & SAVE 25%
SEE PAGE 28 FOR DETAILS

## THIS MONTH:

**14** LEARN TO CODE
Make 2017 a journey of discovery!

**42** REMOTELY ACCESS YOUR PI
Learn how to use SSH in another Raspberry Pi 101 guide

**70** USE DEBIAN + PIXEL ON YOUR PC
Test-drive your Raspberry Pi experience on a spare PC today!

**96** MAKE 2017 A YEAR FOR MAKING
New Year's resolutions don't come any better than this

**FIND US ONLINE** raspberrypi.org/magpi **GET IN TOUCH** magpi@raspberrypi.org

The MagPi

Available on the App Store

GET IT ON Google Play

CC BY NC SA

**EDITORIAL**
Managing Editor: **Russell Barnes**
**russell@raspberrypi.org**
Features Editor: **Rob Zwetsloot**
News Editor: **Lucy Hattersley**
Sub Editors: **Laura Clay, Phil King, Lorna Lynch**

**DESIGN**
Critical Media: **criticalmedia.co.uk**
Head of Design: **Dougal Matthews**
Designers: **Lee Allen, Mike Kay**

**PUBLISHING**
For advertising & licensing:
**russell@raspberrypi.org +44 (0)7904 766523**
Publisher: **Liz Upton**
CEO: **Eben Upton**

**DISTRIBUTION**
**Seymour Distribution Ltd**
2 East Poultry Ave
London
EC1A 9PT | **+44 (0)207 429 4000**

**SUBSCRIPTIONS**
**Select Publisher Services Ltd**
PO Box 6337
Bournemouth
BH1 9EH | **+44 (0)1202 586 848**

**CONTRIBUTORS**
**Sam Aaron, Alex Bate, Johannes Berg, Mike Cook, Gareth Halfacree, Phil King, Simon Long, Sean McManus, Matt Richardson & Richard Smedley**

recycle
When you have finished with this magazine, please recycle it.

This magazine is printed on paper sourced from sustainable forests and the printer operates an environmental management system which has been assessed as conforming to ISO 14001.

The MagPi magazine is published by Raspberry Pi (Trading) Ltd., 30 Station Road, Cambridge, CB1 2JH. The publisher, editor, and contributors accept no responsibility in respect of any omissions or errors relating to goods, products or services referred to or advertised in the magazine. Except where otherwise noted, content in this magazine is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 3.0 Unported (CC BY-NC-SA 3.0). ISSN: 2051-9982.

# Contents

Issue 53 January 2017 raspberrypi.org/magpi

## TUTORIALS


> PI 101 – CONNECT WITH SSH **42**
Learn how to remotely access your Raspberry Pi

> PI 101 – USING THE PI CAMERA **44**
An introduction to the Raspberry Pi Camera Module

> MAKE A GPIO MUSIC BOX **46**
Program push buttons to make different sounds

> INTRODUCTION TO C PART 7 **48**
Learn how to use arrays and strings in C

> CREATE A HORSE RACE GAME **50**
Get yourselves to the derby with Pi Bakery

> SHARE YOUR PI POWER **56**
Use BOINC to donate your Pi's resources to science

> TAKE DEEPDREAM PHOTOS **58**
Use Google's DeepDream with Raspberry Pi

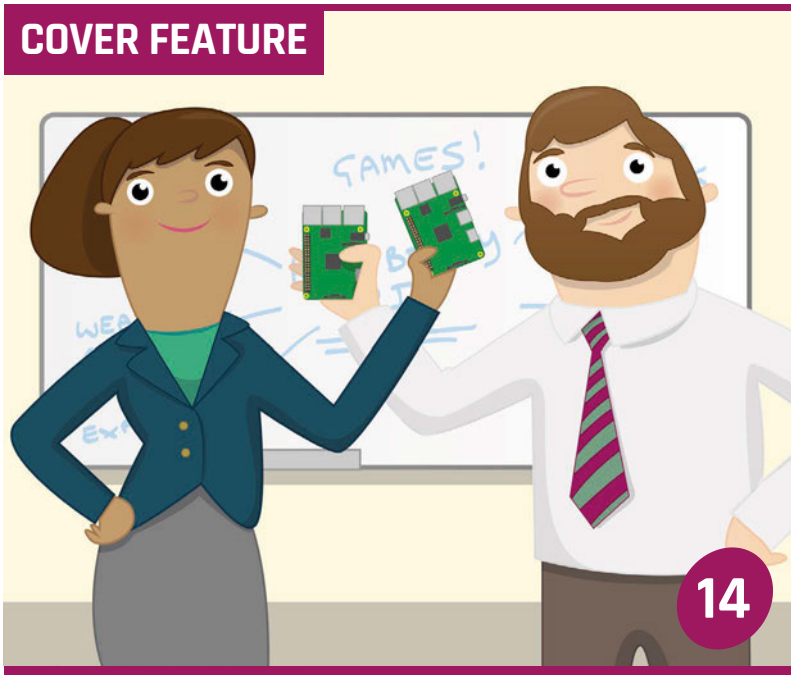
> STRETCH MUSIC SAMPLES **60**
Sonic Pi returns this issue with sample stretching

> RASPBERRY PI SPY CAM **62**
See if anyone's been going into your room

> BOUNCY HEDGEHOG GAME **64**
Put a hedgehog on a trampoline in Scratch!


## IN THE NEWS

## COVER FEATURE




# LEARN TO CODE

## 11 MILLION PIS SOLD

Another milestone for the Raspberry Pi, next stop the C64 **10**

## EBEN HONOURED

Eben Upton receives his CBE after being placed on the honours list in June **10**

# Contents

THE BIG FEATURE

## PIXEL FOR DESKTOPS

Learn how to use the x86 release of Raspbian on your PC, included on our cover disc!

70

## WIN

A MYSTERY MAKER BOX OF GOODIES!

94

## PIONEERS

The teen programme from Raspberry Pi has launched!

11

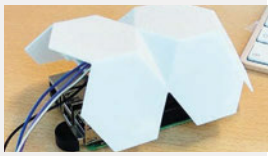
YOUR PROJECTS

30

# PEGASUS

Trying to break the land-speed record with a Pi-powered car


QBEE **34**
A dress that reacts to social media



PI PIPE ORGAN **36**
This classic wooden organ is Pi controlled

AUTOMATED WATER TANK **38**
The PIoT challenge winner shows us his project

## REGULARS


> NEWS **06**
> TECHNICAL FAQ **66**
> BOOK REVIEWS **84**
> FINAL WORD **96**


## COMMUNITY


> THIS MONTH IN PI **86**
Other goings-on in the world of Raspberry Pi
> COMMUNITY SPOTLIGHT **88**
We talk to robot-wrangler Jillian Ogle
> EVENTS **90**
Find a community event near you
> LETTERS **92**
We answer your letters about the magazine and Pi


## REVIEWS


> MIROBOT **78**
> PROTOBOARD **80**
> MOTE PHAT **81**
> LIBREELEC **82**

# THE SOUNDS OF NEW YORK CITY

New York University is using hundreds of Raspberry Pis in the heart of the Big Apple

**Right** New York is one of the busiest cities on Earth, and noise pollution is a big problem for its inhabitants

SONYC
SOUNDS OF NEW YORK CITY

Sounds of New York City is a novel project by New York University (NYU) to monitor noise pollution in the city.

"Noise pollution is one of the topmost quality of life issues for urban residents in the United States," says NYU. "It has been estimated that nine out of ten adults in New York City (NYC) are exposed to excessive noise levels."

To analyse the problem, NYU has set up Raspberry Pi devices with microphones to monitor audio levels around NYC.

We caught up with Charlie Mydlarz, a senior research scientist working on the project.

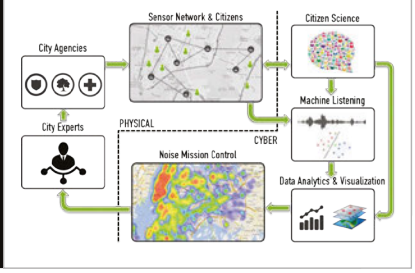
"We are working with NYC agencies, including the Department of Environmental Protection, and we will provide them with an aggregated noise data stream so they can enforce the noise code more efficiently," explains Charlie. "We also work with the Department of Health and Mental Hygiene in a bid to uncover relationships between health and noise in the city."

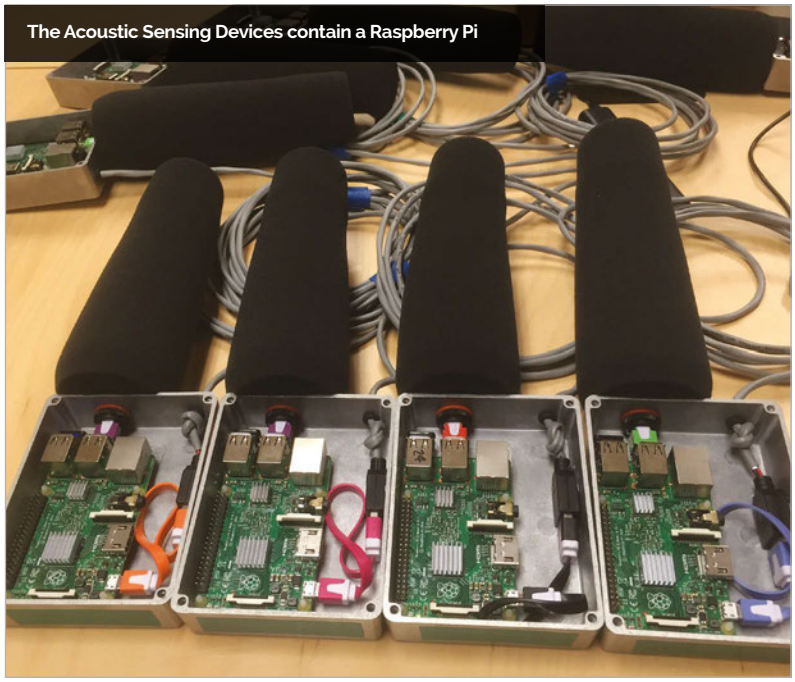
## Sensor nodes

"The sensor node is built around the Raspberry Pi 2B running a headless Jessie," Charlie reveals. Attached to the Raspberry Pi is a digital microphone. "All data is transmitted via WiFi," adds Charlie. It's currently running

## TAKING CONTROL

**The data recorded by the Raspberry Pi are being used to create a Mission Control Centre. Information from the network flows through the infrastructure that analyses the retrieved data. Machine learning techniques are being used to identify sources of noise pollution, such as police sirens or street music.**




The Acoustic Sensing Devices contain a Raspberry Pi

> Noise pollution is one of the topmost quality of life issues for urban residents in the United States

on the NYU network across its campus.

"The sensors themselves don't have a name," he tells us, "but I usually call them Acoustic Sensing Devices or ASDs. 'Listening box' sounds way too ominous!"

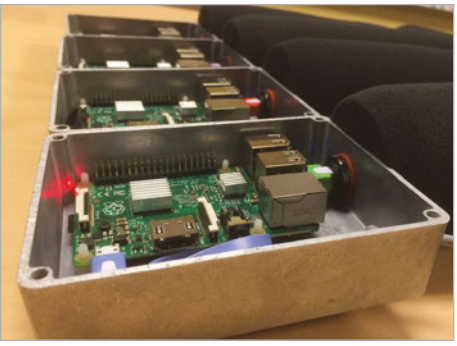
The team has worked hard to ensure that the project doesn't encroach on privacy. "The audio data is collected in ten-second snippets, randomly separated in time to ensure privacy is maintained," says Charlie. These audio snippets are compressed using the lossless FLAC encoder, and then encrypted using AES and RSA encryption.

"We have done a lot of work to maintain privacy on the project, and have had an external consultant confirm that street-level, intelligible speech at conversational levels cannot be picked up," insists Charlie. "A person would have to shout at the sensor for the speech to be intelligible, and that wouldn't constitute a private conversation."

The team also deploy signs below each node to inform people what they are doing.

"As of today, we have 20 nodes up and running," Charlie tells us. By the end of 2016 they should have 50 deployed, and by spring 2017 there will be 100.

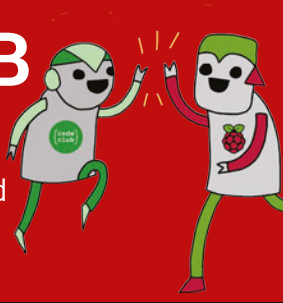
"These poor little Pis have just survived the hot and humid NYC summer and are about to experience the freezing winter," says Charlie. "They've been fantastic so far and have spent weeks running in freezers in the lab, so I'm confident that they can take the winter too."


**Above** New York University plans to deploy over 100 of the devices in its sensor network next year

# THIS MONTH IN NUMBERS

## CODE CLUB

Code Club (codeclub.org.uk) is a nationwide network of after-school clubs that joined forces with The Raspberry Pi Foundation in 2015.



**65,000**
CODE CLUBBERS IN THE UK

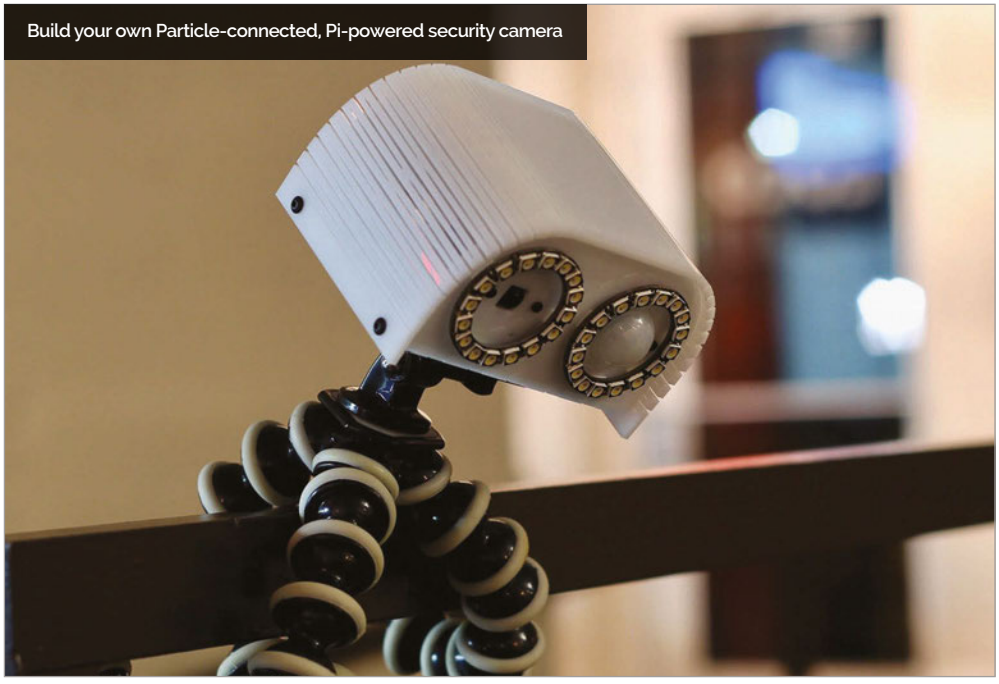
**40%** ARE GIRLS

**4,200**
CODE CLUBS IN PRIMARY SCHOOLS

**470**
IN LIBRARIES AND COMMUNITY CENTRES

# PI JOINS PARTICLE CLOUD

## The world's most popular IoT cloud platform adds Pi support


Build your own Particle-connected, Pi-powered security camera

Particle, a cloud platform for Internet of Things (particle.io), has added official support for the Raspberry Pi. With the popularity of IoT projects on the Pi, the world's most popular low-cost computer seems a natural fit for the world's most popular IoT platform.

"Particle is a scalable, reliable, and secure Internet of Things device platform that enables businesses to quickly and easily build, connect, and manage their connected solutions," comments Jonathan Gladbach, Particle's Head of Growth.

Following a beta programme limited to 1,000 users at the end of November, the service will be made available to all Raspberry Pi owners.

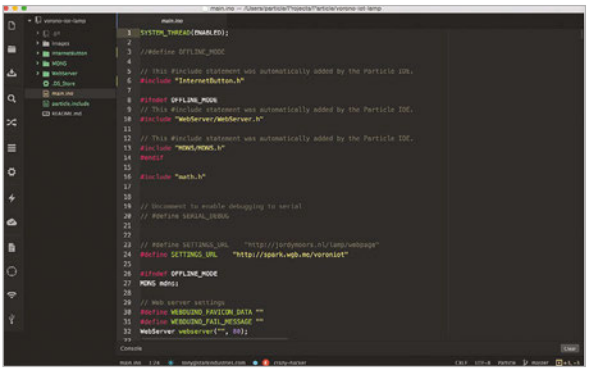
You can sign up for a free account on the website (**magpi.cc/2fuAoM2**) and download the software to your Pi running Raspbian. Using either the Particle Build web IDE or Particle Dev desktop application, you can then select your remote device and program it, with access to numerous community libraries.

"Particle's tools are now used by 70,000 engineers in more than 170 countries," says Jonathan. "[They] are being used by many Fortune 500 companies to develop and manage fleets of new IoT products."

A key feature is the ability to run Arduino Wiring code on the Raspberry Pi: "No more complicated tooling, setup, or scripting to perform simple tasks like trigger a pin, blink an LED, or read a sensor value," explains Jonathan. "With Particle's Raspberry Pi Agent, you can now program in C and C++, making these tasks easy."

Webhooks are tightly integrated into Particle's event system, and projects may be integrated with IFTTT, Google Cloud, or Microsoft Azure.

Use cases for Particle include home automation, remote monitoring, resource management, triggers and buttons, and asset tracking. The Particle website includes Raspberry Pi tutorials (**magpi.cc/2g7kxQB**) to get you up and running with suggested projects, including a security camera (**magpi.cc/2g7uCwH**). Check out Adafruit's demo video, too (**magpi.cc/2g7ndxl**).



**Above** The Particle Dev desktop application allows you to work with local copies of your firmware files

# NEW YEAR. NEW POSSIBILITIES.

## Accessories for Raspberry Pi

### WD SMART CABLE MODULE

The WD Smart Cable Module gives you a compact way to easily integrate a Raspberry Pi Compute Module* with a USB drive to create projects like a wireless NAS or USB camera recorder.

**Upgradeable hardware with built-in CM programming mode**

**2 x USB ports** **10/100 Ethernet jack** **802.11b/g/n Wi-Fi**

*Raspberry Pi Compute Module not included.

$24.99

### MEDIA STICK FOR RASPBERRY PI

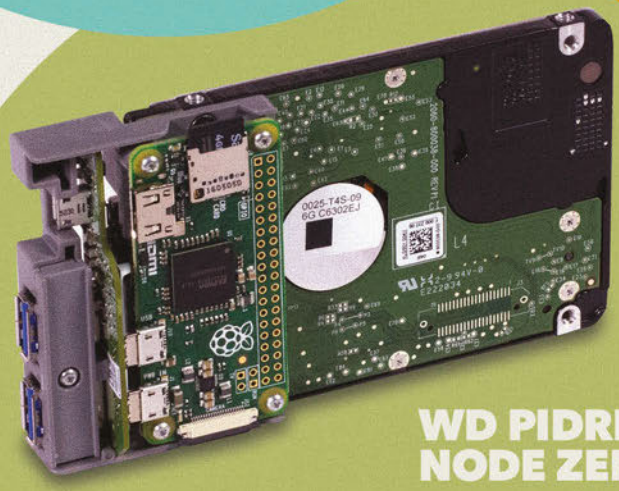
The Media Stick for Raspberry Pi Compute Module* offers an easy way to quickly turn a TV or monitor into a powerful media hub or a custom digital display.

**Upgradeable hardware with built-in CM programming mode**

**1 x HDMI port** **2 x USB ports** **1 x USB Micro B port**

*Raspberry Pi Compute Module not included.

$24.99



### WD PIDRIVE™ NODE ZERO

This all-in-one IoT edge node server offers onboard computing capabilities for autonomous projects like video recording, data logging, and more.

**314GB WD PiDrive connected to Raspberry Pi Zero with custom adapter board**

**SD card preloaded with software**

**Mini HDMI adapter**

$44.99

WDLABS

wdlabs.wd.com/products

# 11 MILLION SOLD

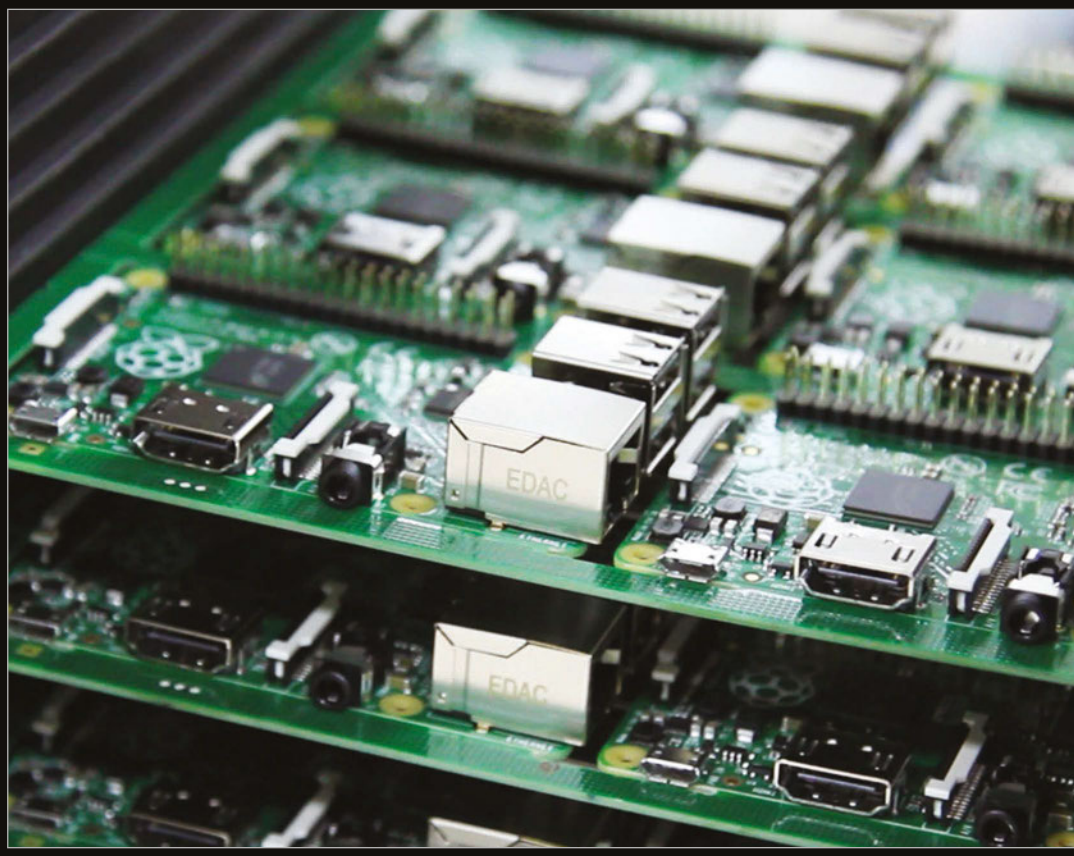
## Raspberry Pi sales pass another major milestone

On 25 November, it was confirmed that total sales of the Raspberry Pi have now topped 11 million.

Considering that Pi sales only reached the ten million milestone at the start of September, that means it's taken less than three months to add a further million.

Even more impressive, it turns out that this figure is only for the flagship Pi models: the B, B+, Pi 2, and Pi 3. "Those are only the $35 product," Raspberry Pi Trading CEO Eben Upton tells us, "so there's no A, no A+, no Compute Module, no Pi Zero." He estimates that sales of the other Pi models total at least another million.

The next milestone will be to bypass sales of the Commodore 64, estimated at 13 million, for the Raspberry Pi to become the third best-selling computer of all time, behind the PC and Mac. "Neither of those are really catchable," reckons Eben.




**Above** Over 11 million Raspberry Pis have been made at the Sony factory in South Wales

# EBEN UPTON CBE

## Raspberry Pi founder is honoured at Buckingham Palace

**Right** Eben Upton CBE says his honour reflects the work of the Raspberry Pi Foundation

Raspberry Pi co-founder Eben Upton has received the CBE (Commander of the Order of the British Empire) medal from HRH Prince Charles during an investiture ceremony on 25 November.

Accompanied by his wife Liz and his parents, Eben described his visit to Buckingham Palace as a "fun day out." He reveals that he and Prince Charles chatted about a previous event they had both attended at Highgrove House a couple of weeks earlier. In addition, "He asked us how many we'd sold. I said 11 million. He said 'Wow!' It looks like someone noticed."

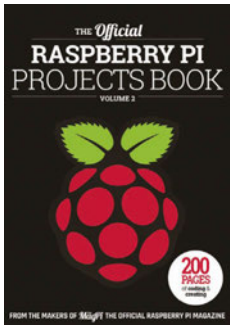
Eben told us he might hang his medal on the Christmas tree. "It certainly would be the prettiest Christmas ornament I've got." While it's a great personal honour, he says it also reflects the work of the Foundation and the many tens of people who have worked for it, along with hundreds of volunteers out in the community.

# NEW OFFICIAL PI PROJECTS BOOK

## Updated Volume 2 includes Pi Zero content

We're proud to announce the release of *The Official Raspberry Pi Projects Book Volume 2*, complete with shiny foil cover. Inside are 200 pages absolutely stuffed with Pi projects, guides, and reviews to help you on your Raspberry Pi journey. Unlike last year's original Projects Book, it includes a ton of Raspberry Pi Zero content to get your teeth into, as well as an updated Getting Started guide.

We know a lot of you love the first *Projects Book* and we really hope you enjoy this new edition of it. You can grab the *Projects Book Volume 2* right now from our online store (**magpi.cc/MagPiStore**) for £12.99, and is available from our app on Android and iOS for a discounted £5.99. It has already made a great stocking filler for Pi enthusiasts, but it's also a great way to start the New Year! It's also available as a free PDF, from **magpi.cc/Back-issues**.




**Right** The latest Pi Projects Book is out just in time for Christmas!

# PIONEERS ARE GO

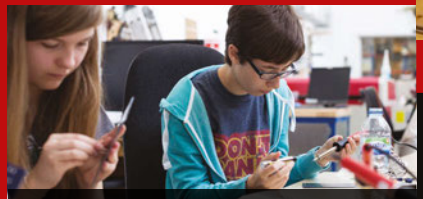
## Young makers can sign up for the challenge

The Raspberry Pi Foundation has officially launched the Pioneers programme (previewed last issue). This means that there's now a lot of extra information available for volunteers and mentors.

Aimed at teenagers, the new programme features a series of challenges to inspire young digital makers. Every few months, Raspberry Pi will set a new mission for the Pioneers community. Each will have a different theme, and the first challenge will be announced in January 2017.

"We want to find and support teenage digital makers in the UK," says Rob Buckland, Director of Programmes. "The aim of Pioneers is to provide guidance, inspiration, and mentorship to teenage makers, and the adults who mentor them."

Young people aged between 12 and 15 will work together in teams, designing and building their ideas to solve the series of challenges. If you fancy giving it a go, or would like to volunteer as a mentor, visit **raspberry.pi/pioneers** and register your interest.



Young digital makers are invited to join the Pioneers programme

# NOW TRENDING

The stories we shared that flew around the world

### HOLA MUNDO!

**magpi.cc/2fv8SMZ**

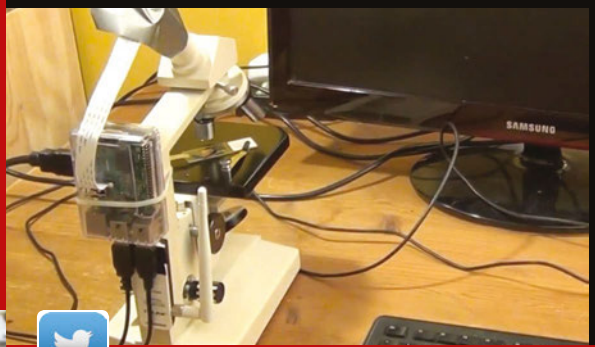
*The MagPi* has been translated into four new languages: French, Spanish, Hebrew, and Italian. Each new magazine is a bite-sized edition, containing 20 pages of the best projects and features.

MISE À JOUR NUMÉRIQUE MENSUELLE GRATUITE DE 20 PAGES
The MagPi
MINI
BRICOLAGE ET CRÉATION AVEC
RASPBERRY PI
PAR LES CRÉATEURS DU MAGAZINE OFFICIEL DU RASPBERRY PI

### ZERO NES

**magpi.cc/2fvdQJF**

Build your own equivalent to Nintendo's NES Classic, using an old game cartridge and a Pi Zero. Snazzy Labs' video showing you how to do it has already racked up well over 200,000 views.




### DIY MICROSCOPE

**magpi.cc/2fv8j5K**

Twitter magnified interest in this DIY digital microscope. The Post Apocalyptic Inventor stuck a Pi custom camera to the eyepiece of a standard scope for a low-cost solution that works amazingly well.

# STAR TREK SERVER

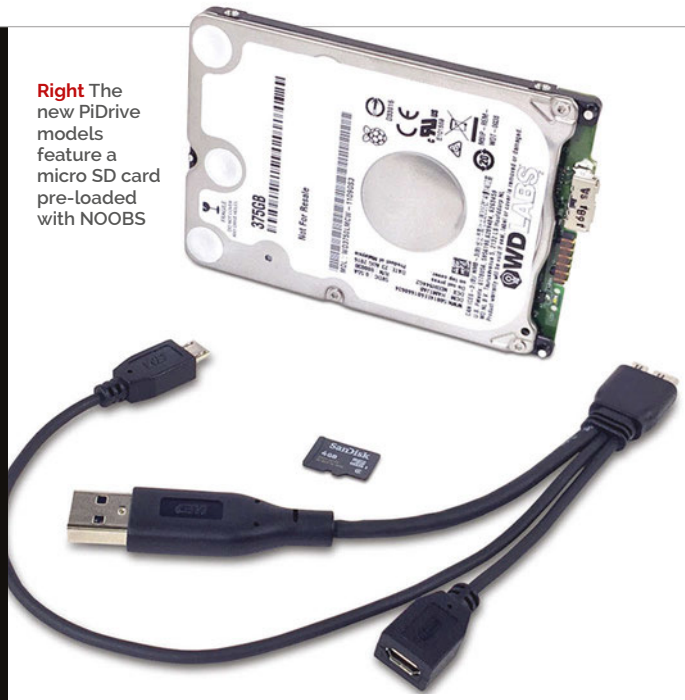
## Sci-fi star beams up a stellar Pi project

Set phasers to stun: Star Trek's very own Wil Wheaton has been building a Raspberry Pi home media server. The American actor, best known for playing Wesley Crusher in *Star Trek: The Next Generation*, was 'hailing all frequencies' on Twitter, **@wilw**, to announce the news to fans: "I'm having so much fun with my Raspberry Pi weekend project." Following an online tutorial by developer Mel Grubb (**magpi.cc/2fIs36c**), Wil is using a Raspberry Pi 3 running Raspbian, along with a hard drive, to 'make it so'.

Asked by a Twitter follower whether it could boost the efficiency of the warp core, Wil responded, "If you add the right repository to apt.sources, it can." He also revealed that he's finding it hard to resist getting a Picade retro gaming system: "It's getting increasingly difficult." Well, as most retro games fans already know, 'resistance is futile'.

Wil Wheaton is hoping his Pi-powered home server will live long and prosper

© Luigi Novi / Wikimedia Commons



**Right** The new PiDrive models feature a micro SD card pre-loaded with NOOBS

# WD PIDRIVE FOUNDATION EDITION

## Featuring NOOBS preloaded on SD

WDLabs has launched the third generation of its PiDrive products, offering an affordable hard drive storage solution for the Raspberry Pi. Known as the 'Foundation Edition', the new range includes two hard drive options – 250GB and 375GB – along with a 64GB USB flash drive.

While similar in format to previous PiDrive models, this edition features a micro SD card preloaded with the Raspberry Pi Foundation's NOOBS installer, enabling users to easily install Raspbian with PIXEL, or an alternative operating system, without the need to flash a memory card manually.

> "WDLabs has launched the third generation of its PiDrive products"

"This third generation WD PiDrive solution uses a USB HDD or USB flash drive to run the OS and host multiple Raspberry Pi projects, instead of having to do this on a collection of micro SD cards," says Dave Chew, chief engineer at WDLabs.

Available directly from **wd.com**, the PiDrive Foundation range is priced from $18.99 to $37.49.

# SECURITY UPDATE FOR RASPBIAN

## Latest version of Raspbian OS beefs up security

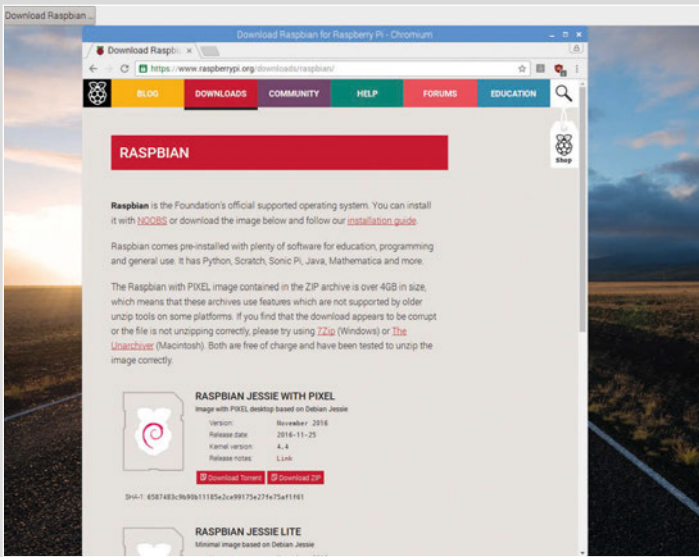
The Raspberry Pi Foundation released a new version of the Raspbian operating system on 25 November 2016.

The 1.1 release of Raspbian with PIXEL includes a number of fixes and adjustments. The new version of Raspbian also introduces a range of security measures.

"We usually find a few small bugs and other issues as soon as the wider community start using it, and so we gather up the fixes and produce a 1.1 release a few weeks later," writes Simon Long on The Raspberry Pi Foundation's blog (**magpi.cc/2gF9mhY**).

You can download the new image file from the Foundation's Downloads page at **magpi.cc/1MYYTMo**. Or update your existing Jessie OS using these commands in a terminal:

```
sudo apt-get update
sudo apt-get dist-upgrade
sudo apt-get install -y pprompt
```




TAKING THE RASPBERRY PI TO THE NEXT LEVEL

HIPSY
HOME INTRUSION PREVENTION SYSTEM

24 HOUR

GIPSY
GLOBAL INTRUSION PREVENTION SYSTEM

MANAGED CYBER-PROTECTION SERVICE

DEVELOPED BY THE INDUSTRY EXPERTS
idappcom

LINUX OPERATING SYSTEM • INDUSTRY STANDARD INTRUSION DETECTION ENGINE • SMALL WEB SERVER • DNS SERVER • NETWORK SCANNER
ENHANCED WIRELESS FUNCTIONS • CLOUD BASED MANAGEMENT VPN • IDAPPCOM RULES MANAGEMENT SYSTEM
ACCESS TO 10,000 PLUS EXPLOIT RULES • OPTIONAL MALWARE AND PHISHING PROTECTION

PROTECT YOUR FAMILY, FINANCES & FILES AT HOME & ON THE GO

SHIELD FROM EXPLOITS
Attackers working from outside your system

PROTECT FROM PHISHING
Invitations to reveal your private information

BLOCK MALWARE
Attacks that have been installed inside your system

Our products are built on the best of British small computer, using our decades of expert security experience. We provide 24 hour security management via the cloud connected Idappcom Security Operations Centre.

Taking the Raspberry Pi to the Next Level of Cyber Protection
Visit: www.ipssecurityrules.com Call: +44 (0) 203 355 6804

# Beginner's Guide to CODING

## Discover the joy and art of computer programming with your Raspberry Pi

Learning to code is one of the most profoundly life-changing things you can do. This has always been true, but learning to code is increasingly important in the modern world.

The reason the Raspberry Pi was created was to challenge a drop in computer science applications at Cambridge University. Modern computers, and especially games consoles, were fun and powerful, but not easily programmable.

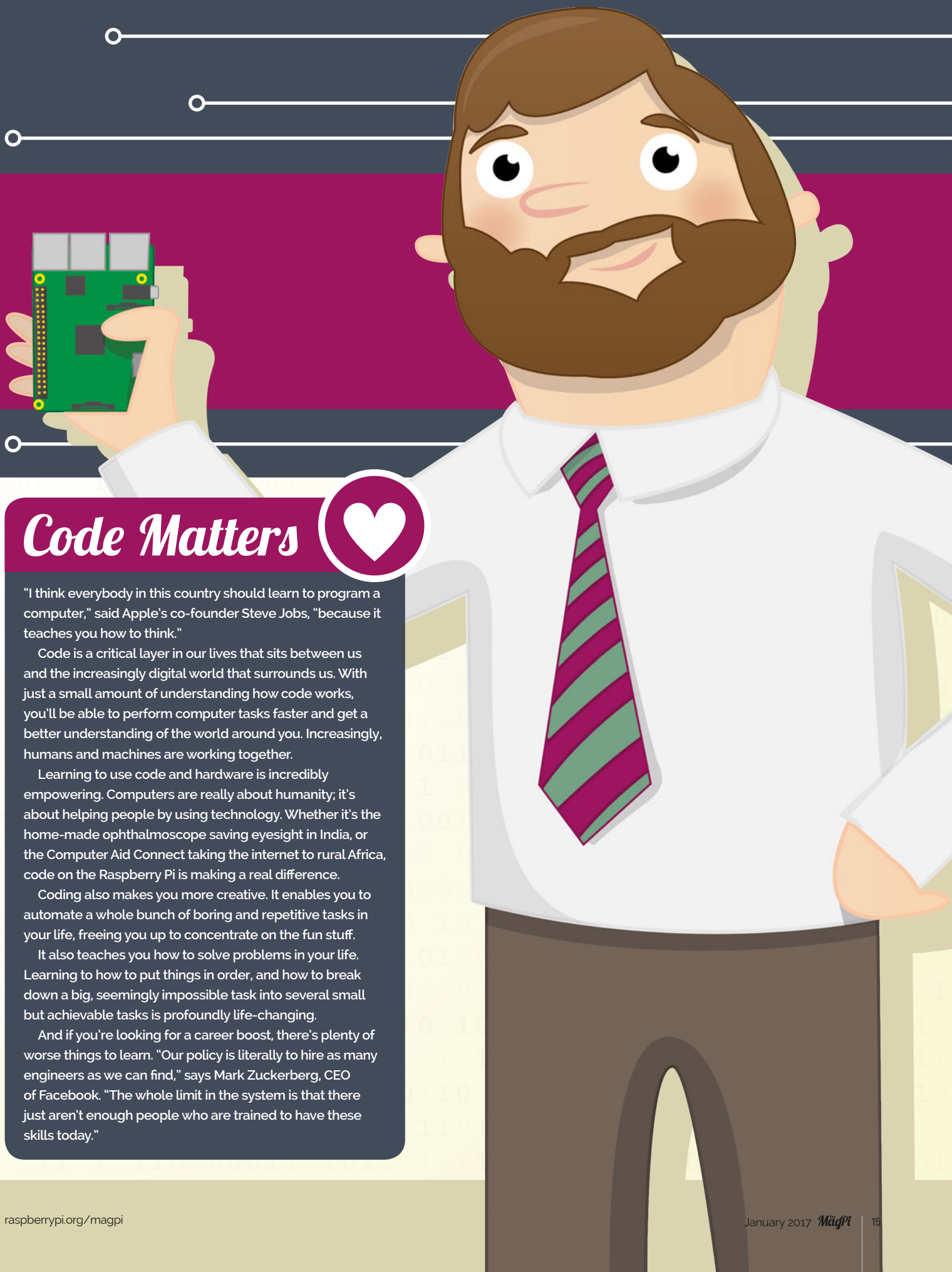
The maker community fell in love with the Raspberry Pi, thanks to its cheap and hackable nature. Building and tinkering are the primary reasons we love Raspberry Pi. Great projects use a combination of hardware and software together.

So, whether you're a hacker learning to make better projects, or a would-be coder looking for a better career, this feature is set to help you on your way.

The good news is that you don't need to be a genius to know coding, just as you don't have to be a genius to read and write. It's actually pretty simple once you learn a few simple concepts like variables, branching, and loops.

Perhaps you're brand-new to coding. Maybe you did a little BASIC in school, or used old languages like Pascal and Fortran. Or maybe you're already knee-deep in projects and just want to learn the language that controls them.

Wherever you're coming from, we're here to walk you through the basic concepts of computer programming. We'll demystify the whole process of code, so you can get a better understanding of what's going on inside your Raspberry Pi.



# Code Matters

"I think everybody in this country should learn to program a computer," said Apple's co-founder Steve Jobs, "because it teaches you how to think."

Code is a critical layer in our lives that sits between us and the increasingly digital world that surrounds us. With just a small amount of understanding how code works, you'll be able to perform computer tasks faster and get a better understanding of the world around you. Increasingly, humans and machines are working together.

Learning to use code and hardware is incredibly empowering. Computers are really about humanity; it's about helping people by using technology. Whether it's the home-made ophthalmoscope saving eyesight in India, or the Computer Aid Connect taking the internet to rural Africa, code on the Raspberry Pi is making a real difference.

Coding also makes you more creative. It enables you to automate a whole bunch of boring and repetitive tasks in your life, freeing you up to concentrate on the fun stuff.

It also teaches you how to solve problems in your life. Learning to how to put things in order, and how to break down a big, seemingly impossible task into several small but achievable tasks is profoundly life-changing.

And if you're looking for a career boost, there's plenty of worse things to learn. "Our policy is literally to hire as many engineers as we can find," says Mark Zuckerberg, CEO of Facebook. "The whole limit in the system is that there just aren't enough people who are trained to have these skills today."

# What is a Program?

Discover the building blocks of software and learn what goes on inside a program

## Which Python?

Python 2 and Python 3 are both commonly used. Python 3 is the future, so we're going with it. Lots of courses still teach Python 2, and it's not a bad idea to take a closer look at the differences between the two: **magpi.cc/2gP6zX3**

Before you go any further, let's look at what a program actually is. The dictionary definition is a "set of instructions that makes a computer do a particular thing."

A computer program is a lot like a recipe. It has a list of ingredients, called 'variables', and a list of instructions, known as 'statements' or 'functions'. You follow the instructions from the recipe one line at a time and end up with a tasty cake – and that's no lie.

The real miracle of computers, however, is that they can do the same thing repeatedly. So you can get a machine to bake a thousand cakes without ever getting tired. A program may contain loops that make it do the same thing over and over again.

Programs also make decisions, and different paths through a program can be taken. Your recipe could make a scrummy chocolate cake or a delightful batch of doughnuts, depending on the variables (the ingredients) it has.

## Compiled vs Interpreted

Python is an 'interpreted language'. You write the code and then run the program. Under the hood, it's being translated and runs on the fly. Some languages, such as C and Java, are compiled. You write the program, then compile it to get a build file (written in machine code), then you run the build. It's a faff you can do without for now.

One thing that may surprise you when you begin programming is just how little you need to know to get started. With just a few variables, a smattering of flow, and some functions, you can get a computer doing all the hard work for you.

## Inside your Pi

At the heart of your Raspberry Pi are billions of voltage switches known as binary digits (or 'bits' for short). There are 8,589,934,592 of them in its 1GB of RAM, to be exact. All these switches can be set to high or low, which is typically represented as 0 (for low or off) and 1 (for high or on). Everything you see on the screen, hear from the speakers, and type on the keyboard is billions of switches being turned on and off.

Obviously, it's not that easy for humans to talk directly to computers. It's possible to use machine language and send binary instructions directly to a computer, but this isn't where any sane person starts (or ends if they want to remain sane).

Instead, we use a coding language to program. This is written using easy-to-understand functions like `print()`. These are then interpreted into machine language, which the computer understands.

We're going to use Python to learn to code. Python is a truly great programming language. It has a rich syntax that's free from clutter; you don't have to worry about things like curly braces and static typing that crop up in more complicated languages like Java.

With Python, you can just create code, run it, and get things done. Python is one of the languages found most commonly inside *The MagPi*, so learning it here will help you understand lots of the code used in projects.

# IDE and IDLE

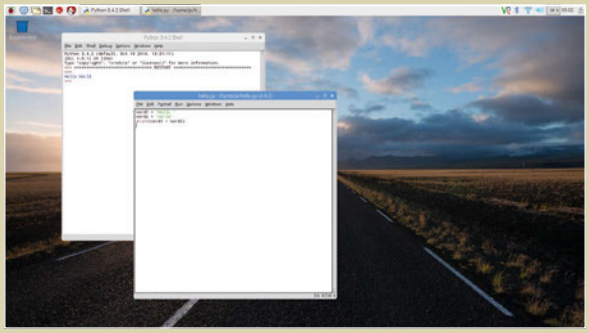
You don't have to write Python programs using a text editor like Leafpad and run them in the terminal. Instead, you can use a neat all-in-one solution, known as an 'IDE' (integrated development environment).

IDEs combine a text editor with program-running functionality. Often, they'll include fancy features like debugging and text completion.

Click **Menu > Programming > Python 3 (IDLE)**, and you'll get a new window called 'Python 3.4.2 Shell:'. This Shell works just like Python on the command line. Enter `print("Hello World")` to see the message.

You can also create programs in a built-in file editor. Choose **File > New File**. Enter this program in the window marked 'Untitled':

```
word1 = "Hello "
word2 = "World"
print(word1 + word2)
```



**Above** Python IDLE makes it easy to create programs and run them without having to use the command line

Don't forget to include the space after 'Hello'. Choose **File > Save As** and save it as **hello.py**. Now press **F5** on your keyboard to run the program. (Or choose **Run > Run Module**). It'll display 'Hello World' in the Shell.

The advantage of using Python IDLE is that you can inspect the program in the Shell. Enter `word1`, and you'll see 'Hello '. Enter `word2` and you'll see 'World'. This ability to inspect and use the variables in your program makes it a lot easier to experiment with programming and detect bugs (problems in your code).

## Why Python?

There are a lot of programming languages out there, and they all offer something special. Python is a great option for beginners. Its syntax (the use of words and symbols) is easy to read. And it scales all the way up to industrial, medical, and scientific purposes, so it's ideal for beginners and experts alike.

## Python in the terminal

You don't need to do anything to set up Python on your Raspberry Pi. Open a terminal in Raspbian and enter `python --version`. It will display 'Python 2.7.9'. Enter `python3 --version` and you'll see 'Python 3.4.2'.

We're going to use Python 3 in this feature (see 'Which Python?' boxout). You can open Python 3 in the terminal by just typing `python3`.

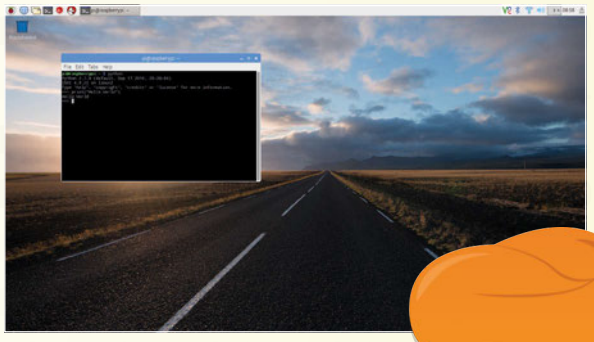
The '$' command-line prompt will be replaced with '>>>'. Here you can enter Python commands directly, just as you would terminal commands.

It's tradition to christen any new language by displaying 'Hello World'. Enter `print("Hello World")` and press **RETURN**. You'll see 'Hello World' outputted on the line below.

Using the Shell is known as Interactive Mode. You can interact directly with the code. It's handy for doing maths; enter `1920 * 1080` to get the answer: 2073600.

Mostly, you create Python programs using a regular text editor and save the files with a '.py' extension. Don't use a word processor like LibreOffice Writer, though – it'll add formatting and mess up the code.

Use a plain text editor like Leafpad (**Menu > Accessories > Text Editor**). Here you can enter your code, save it as a program, and then run the file in the terminal. Enter `python3 yourprogram.py` at the command line to run a program.



**Left** Python comes pre-installed in the Raspbian operating system and you can use it at the command line

# Variables

**Variables are all-purpose containers that you use to store data and objects**

## Python types

Python has five standard data types:

- Numbers
- String
- List
- Tuple
- Dictionary

## Foo bar?

You'll come across 'foo' and 'bar' a lot when learning to code. These are dummy placeholders and don't mean anything. They could be zig and zag or bim and bam. Nobody's quite sure, but it might be related to the expression 'fubar' from the Vietnamese war.

If you've created a science project or experiment, you may have come across variables. In science, a variable is any factor that you can control, change or measure.

In computer programming, variables are used to store things in your program. They could be names, numbers, labels, and tags: all the stuff your program needs.

In Python, you write the name of a variable then a single equals sign and the word, number or object you want to put in it.

Enter this code directly into the Shell:

```
foo = 1
bar = 2
```

Remember: the variable name is on the left, and the thing it contains is on the right. Imagine you've got two plastic cups, and you've scrawled 'foo' on the first and 'bar' on the second. You put a number 1 in foo and a number 2 in bar.

If you ever want to get the number again, you just look in the cup. You do this in Python by just using the variable name:

```
foo
bar
```

You can also print out variables by passing them into a **print** function:

```
print(foo)
print(bar)
```

Variables can also be used to contain 'strings'. These are groups of letters (and other characters) that form words, phrases or other text.

Creating a string variable in Python is pretty much the same as creating an integer, except you surround the text with single (**' '**) or double (**" "**) quotes.

Using double quotes makes it easier to include apostrophes, such as **print("Don't worry. Be Happy")**. This line would break after 'Don' if you used single quotes – **print('Don't worry, be happy')** – so use double quotes for now.

## Why variables count

Variables make it much easier to change parts of your code. Say you've got an excellent coding job at Nursery Rhymes Inc and you've written a classic:

```
print("Polly put the kettle on")
print("Polly put the kettle on")
print("Polly put the kettle on")
print("We'll all have tea")
```

The head of marketing comes in and says "our data shows that Polly isn't trending with the millennial demographic." You say "Huh!" and he barks "Change Polly to Dolly."

You now have to go through and change the variable in all three lines. What a downer! But what if you'd written thousands of lines of code and they all needed to change? You'd be there all week.

With variables, you define the variable once and then use it in your code. Then it's ready for changing

at any time:

```
name = "Polly"

print(name + " put the kettle on)
print(name + " put the kettle on)
print(name + " put the kettle on)
print("We'll all have tea")
```

This code prints out the same classic nursery rhyme. But if you want to change the name of our character, you only have to change it in one place:

```
name = "Dolly"
```

…and the poem will update on every line.

## What's your type?

When you create a variable in Python, it's automatically assigned a type based on what it is. You can check this using the **type()** function. In the shell interface, enter:

```
foo = "Ten"
bar = 10
```

Now use the **type()** function to check the type of each variable:

```
type(foo)
type(bar)
```

It will say **<class 'str'>** for **foo**, and **<class 'int'>** for **bar**. This concept is important, because different types work together in a variety of ways, and they don't always play nicely together.

For example, if you add together two strings they are combined:

```
name = "Harry"
job = "Wizard"
print("Yer a " + job + " " + name)
```

This prints the message "Yer a Wizard Harry". The strings are concatenated (that's a fancy programming term for 'joined up'). Numbers, though, work completely differently. Let's try a bit of maths:

```
number1 = 6
number2 = 9

print(number1 + number2)
```

Instead of concatenating 6 and 9 together to give 69, Python performs a bit of maths, and you get the answer '15'.

## Type casting

So what happens when you want to add a string and an integer together?

```
name = "Ben"
number = 10
print(name + number)
```

You'll get an error message: 'TypeError: Can't convert 'int' object to str implicitly'. This error is because Python can't add together a string and an integer, because they work differently. Ah, but not so fast! You can multiply strings and integers:

```
print(name * number)
```

It'll print 'Ben' ten times: you'll get 'BenBenBenBenBenBenBenBenBenBen'.

If you want to print out 'Ben10', you'll need to convert the integer to a string. You do this using a **str()** function and putting the integer inside the brackets. Here we do that, and store the result in a new variable called **number_as_string**:

```
number_as_string = str(number)
print(name + number_as_string)
```

This code will print out the name 'Ben10'. This concept is known as 'type casting': converting a variable from one type to another.

You can also cast strings into integers using the **int()** function. This is particularly useful when you use **input()** to get a number from the user; the input is stored as a string. Let's create a program that asks for a number and exponent and raises the number to the power of the exponent (using the '**' symbol):

```
number = input("Enter a number: ")
exponent = input("Enter an exponent: ")
result = int(number) ** int(exponent)
```

Our first two variables, **number** and **exponent**, are strings, while our third, **result**, is an integer. We could just print out the result:

```
print(result)
```

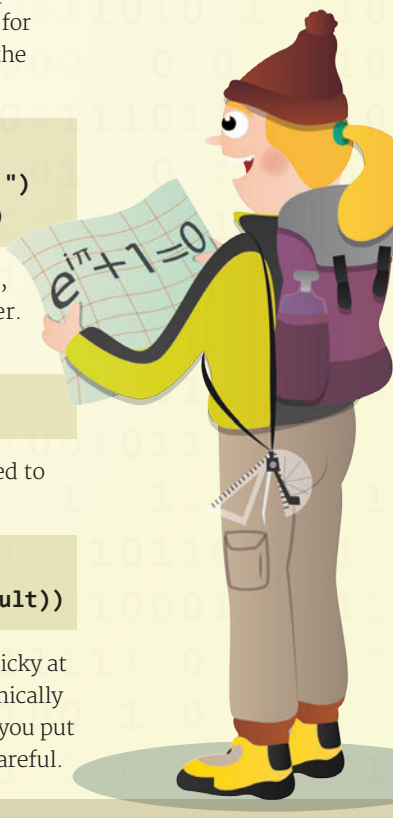
But if we wanted to include a message, we need to type cast **result** to a string:

```
print(number + " raised to the power
      " + exponent + " is " + str(result))
```

Variables, types, and type casting can be a bit tricky at first. Python is a lot easier to use because it dynamically changes the type of a variable to match the thing you put in it. However, it does mean you have to be a bit careful.

## What to call a variable?

Variable names should be lower-case words separated by an underscore '_'. They can include numbers, but must start with a letter. You can call variables pretty much anything, but there's a small list of reserved keywords you should avoid (**magpi.cc/2h7MH1y**). It's a good idea to call them something that will be obvious when you use them in your program, like 'student_name' or 'person_age'.



# Controlling flow with While & For

Get your program to do all the hard work with while and for loops

## Comparison operators

These comparison operators are commonly used in conditions to determine if something is True or False:

| | |
|---|---|
| == | equal |
| != | not equal |
| < | less than |
| <= | less than or equal to |
| > | greater than |
| >= | greater than or equal to |
| <> | less than or greater than |

Computers are great because they don't mind doing the same stuff over and over again. Their hard-working nature makes computers ideal for doing grunt work.

When looking at variables earlier, we printed out this nursery rhyme:

```
print("Polly put the kettle on")
print("Polly put the kettle on")
print("Polly put the kettle on")
print("We'll all have tea")
```

We didn't like the repetition of Polly, so we replaced it with a variable. But this code is foolish in another way: you have to write out the same **print** line three times.

We're going to use a loop to get rid of the repetition. The first loop we're going to look at is a 'while loop'. In Python 3 IDLE, create a new file and save it as **polly.py**; enter the code from the top of the next page.

We start with two variables:

```
name = "Polly"
counter = 0
```

Then we use the **while** statement followed by a condition: **counter < 3**.

On the next line down, you press the space bar four times to indent the code. Don't press the **TAB** key (see 'Tabs or spaces?' boxout).

```
while counter < 3:
    print(name + " put the kettle on")
    counter = counter + 1
```

The **<** symbol stands for 'less than'. It checks if the item on the left is less than the item on the right. In this case, it sees if the variable counter (which starts at 0) is less than 3. This condition is known as 'True'; if it wasn't, it'd be known as 'False'.

Finally, enter the last line of code:

```
print("We'll all have tea")
```

Save and run the program (press **F5**). It will print 'Polly put the kettle on' three times and then 'We'll all have tea'.

### While, condition and indent

There are three things here: the while statement, the condition, and the indented text, organised like this:

```
while condition:
    indent
```

Imagine a three-way chat between all three items in our **polly.py** program:

## Tabs or spaces?

There's a massive nerd debate about whether to use spaces or tabs when indenting code. There are valid arguments on both sides, which you can learn in this clip from HBO's comedy *Silicon Valley* (**magpi.cc/2gZde0M**). Use spaces for now. When you're a hardcore coder, you can make the argument for tabs.

**While:** "Hey Condition! What's your status?"
**Condition:** "True! The counter is 0. It's less than 3."
**Indent:** "OK, guys. I'll print out 'Polly put the kettle on' and increase the counter by 1. What's next?"

**While:** "Hey Condition. What's your status?"
**Condition:** "True! The counter is now 1."
**Indent:** "OK. I'm printing out another 'Polly put the kettle on' and increasing the counter by 1."

This goes on till the counter hits 3.

**While:** "Hey Condition. What's your status?"
**Condition:** "False! The counter is now 3, which isn't less than 3."
**While:** "OK guys. We're done!"

The program doesn't run the indented code, but moves to the single **print** at the end: 'We'll all have tea'.

## For and lists

The next type of loop is known as 'for'. This is designed to work with lists.

Lists are a type of variable that contain multiple items (strings, numbers, or even other variables). Create a list by putting items inside square brackets:

```
banana_splits = ["Bingo", "Fleegle",
                 "Drooper", "Snorky"]
```

Now enter **banana_splits** in the Shell to view the list. It will display the four names inside the square brackets. You can access each item individually using the variable name and square brackets. Enter:

```
banana_splits[0]
```

...and you'll get 'Bingo'. Lists in Python are zero-indexed; that means the first item in the list is [0]. Here are each of the items. Type them into the Shell to get the names returned:

### Polly.py

```
name = "Polly"
counter = 0

while counter < 3:
    print(name + " put the kettle on")
    counter = counter + 1

print("We'll all have tea")
```

```
banana_splits[0] # "Bingo"
banana_splits[1] # "Fleegle"
banana_splits[2] # "Drooper"
banana_splits[3] # "Snorky"
```

Zero-indexed lists can be confusing at first. Just remember that you're counting from 0. A for loop makes it easy to iterate over items in a list. Create this program and save it as **splits.py**:

```
banana_splits = ["Bingo", "Fleegle",
                 "Drooper", "Snorky"]

for banana_split in banana_splits:
    print(banana_split)
```

It doesn't matter what you use as the variable in a for loop, as long as you remember to use it in your indented code. You could put:

```
for dude in banana_splits:
    print(dude)
```

It's common to name the list as something plural (such as 'names', 'pages', and 'items') and use the singular version without the 's' for the 'in' variable: 'for name in names', 'for page in pages', and so on.

### Infinite loops

You must be careful to change the counter in a while loop, or you'll get an infinite loop. If you delete the line **counter = counter + 1** from our while loop, it will run forever: it never goes above 0, so the indented code runs over and over again. This bug is known as an 'infinite loop' and is a bad thing to have in your programs.

# Conditional Branching

Give your programs some brains with conditional branching

## Logical operators

You can combine conditions together using logical operators.

**and** Both operands are true: (a and b) is True

**or** Any operator is true: (a or b) is True

**not** Checks if something is false: not (a and b) is True if both a and b are False.

Your programs have been slowly getting more powerful. We've learned to run instructions in procedural order, replaced parts of our program with variables, and looped over the code.

But another important part of programming is called 'conditional branching'. Branching is where a program decides whether to do something or not.

Of course, a program doesn't just decide whether or not to do things on a whim: we use the sturdy world of logic here.

The start of all this is the powerful 'if' statement. It looks similar to a loop, but runs just once. The if statement asks if a condition is True. If it is, then it runs the indented code:

```
if True:
    print("Hello World")
```

Run this program, and it'll display 'Hello World'. Now change the if statement to False:

```
if False:
    print("Hello World")
```

...and nothing will happen.

Of course, you can't just write True and False. Instead, you create a condition which evaluates to True or False; a common one is the equals sign (==). This checks whether both items on either side are the same. Create a new file and enter the code from **password1.py**. This code is a simple program that asks you to enter a password; if you enter the correct password, 'qwerty', it displays 'Welcome'.

Be careful not to confuse the equals logic operator == with the single equals sign =. While the double equals sign checks that both sides are the same, the single equals sign makes both sides the same. Getting == and = mixed up is a common mistake for rookie coders.

## What else

After if, the next conditional branch control you need to learn is 'else'. This command is a companion to if and runs as an alternative version. When the if branch is True, it runs; when the if branch is False, the else branch runs.

```
if True:
   print("The first branch ran")
else:
   print("The second branch ran")
```

Run this program and you'll see 'The first branch ran'. But change True to False:

```
if False:
   print("The first branch ran")
else:
   print("The second branch ran")
```

...and you'll see 'The second branch ran'. Let's use this to expand our password program. Enter the code from **password2.py**.

Run the program again. If you get the password correct now, you'll get a welcome message. Otherwise, you'll get an 'incorrect password' message.

## Elif

The third branching statement you need to know is 'elif'. This statement stands for 'else if', and sits between the if and else statements. Let's look at an elif statement. Enter this code:

```
if False:
    print("The first block of code ran")
elif True:
    print("The second block of code ran")
else:
    print("The third block of code ran")
```

Run this program and you'll find it skips the first if statement, but runs the elif statement. You'll get 'The second block of code ran'.

The else statement doesn't have a True or False condition; it runs so long as neither the if or elif statements are True. (Note that the else statement here, as always, is optional; you can just have if and elif.)

But what happens if you change both the if and elif conditions to True? Give it a try and see whether just if runs, or elif, or both. Experiment with removing the else statement and play around. It'll help you get the hang of the if, elif, and else statements.

## FizzBuzz

We're going to show you a common program used in computer programming interviews. It's a classic called 'FizzBuzz', and it shows that you understand if, else, and elif statements.

First, you need to know about the modulo operator (**%**). This is used to get the remainder from a division and is similar to a divide operator. Take this function:

```
10 / 4 == 2.5
```

If we use a modulo instead, we get this:

```
10 % 4 == 2
```

Modulo turns out to be handy in lots of ways. You can use **% 2** to figure out if a number is odd or even:

```
10 % 2 == 0 # this is odd
11 % 2 == 1 # this is even
```

This program works out if a number is odd or even:

```
number = 10

if number % 2 == 0:
    print("The number is even")
else:
    print("The number is odd")
```

OK – let's move on to FizzBuzz.

**Password.py**

```
password = "qwerty"
attempt = input("Enter password: ")

if attempt == password:
    print("Welcome")
```

**Password2.py**

```
password = "qwerty"
attempt = input("Enter password: ")

if attempt == password:
    print("Welcome")
else:
    print("Incorrect password!")
```

## Writing FizzBuzz

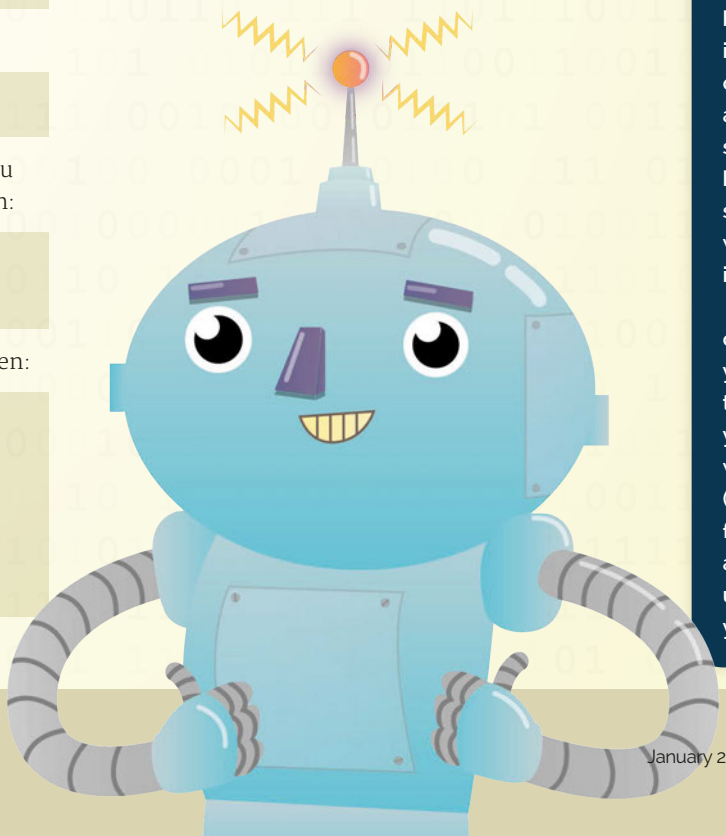
The brief for our FizzBuzz is to print the numbers from 1 to 100. If a number is divisible by three (such as 3, 6, and 9), then you print 'Fizz' instead of the number; if the number is divisible by five, you print 'Buzz' instead.

But if a number is divisible by both 3 and 5, such as the number 15, then you print 'FizzBuzz'.

We're also introducing a new element in FizzBuzz: the 'and' statement. This checks if two conditions are both True: that the number can be divided by both 3 and 5. It only returns True if both conditions are True.

There are three main logical operators: and, or, and not. The first two are relatively straightforward, but the 'not' operator can be more confusing at first. Don't worry about it too much; you'll get the hang of it with practice.

Enter the **fizzbuzz.py** code from page 25 to practise using if, else, and elif elements and logical operators.



## Comments

A mark of a good programmer is to use comments in your programs. Comments are used to explain bits of your program to humans. They are completely ignored by the computer.

In Python, you start a comment line with a hash symbol (#). It can be on a line on it own, or it can come right after a line of code. As soon as Python hits the #, it'll stop translating whatever follows into machine code.

Comments help other users to read your program, but they will also help you understand what you're doing (long after you've forgotten). It's a good habit to use comments in your programs.

# Creating Functions

## Create the building blocks of code and make more robust programs

You've come a long way since your first 'Hello World'. Your programs now check for conditions and loop over themselves.

You're now writing programs that are known as 'Turing complete', named after Alan Turing, the father of computer science and artificial intelligence, who hacked the German Enigma code in WWII.

Now we're going to take things a little further. We're going to introduce you to a form of modularity called functions.

Functions are blocks of code that you write once and can repeat anywhere. It's a little like being able to write a block of text once, and then paste it whenever you need it.

### Spotting a function

Python is packed with built-in functions, and you've already been using them in your programs. Commands like **`print()`**, **`len()`**, and **`type()`** are all functions. They're easy to spot: a small command starting with a lower-case letter and followed by a pair of parentheses '()'.

### Python documentation

You can browse or download a copy of the Python documentation directly from the Python website at **python.org/doc**. Python has a whole bunch of built-in functions. You can view a list of all the built-in functions on the Python documentation website (**magpi.cc/2gPsGK3**).

### Using functions

Let's take a look at a function called **`abs()`**. It stands for 'absolute', and returns the absolute value of any number you pass into it (the bit you pass in is called the 'argument').

An absolute number is the positive of any number, so if you write **`abs(-2)`** you get 2 back. Try this in the Shell:

```
abs(2) # returns 2
abs(-2) # returns 2
```

You can store the returned result as a variable:

```
positive_number = abs(-10)
```

We find it easier to read a function backwards, from right to left. The value is passed into the parentheses, then the function cranks it and returns a new value. This is passed left and stored into the variable.

### Defining a function

The great thing about Python is that you don't just use the built-in functions: you get to make your own. These are called 'user-defined functions'.

You create a function using the **`def`** keyword, followed by the function name and parentheses. Inside the parentheses, you list the parameters. These are the same as the arguments, only inside the definition they are called 'parameters'.

```
def function(parameter):
    return parameter
```

Our function here doesn't do anything; it simply accepts a parameter and returns it.

At the end of the function definition is a colon (**:**). The function code is indented by four spaces, just like a loop or if/else branch.

The code inside the indentation runs when you call the function. Functions typically include a **return** statement which passes back an expression.

## Working functions

We're going to create a function that prints the lyrics to Happy Birthday.

Type out the **happy_birthday.py** code from the listing, then run it. In the Shell, enter:

```
happy_birthday("Lucy")
```

This function call uses the string 'Lucy' as the argument. This string is passed into the function as the parameter and is then available for use in the indented code inside the function.

## Return statements

Many functions don't just run a block of code; they also return something to the function call.

We saw this in **abs()**, which returned the absolute value of a number. This can be stored in a variable.

In fact, we're going to recreate the **abs()** function, so you can see how it's working behind the scenes.

In maths, you invert a positive/negative value by multiplying a negative number by -1, like this:

```
10 * -1 = -10
-10 * -1 = 10
```

We need to create a function that takes a number as a parameter and checks if it's negative. If so, it multiplies it by -1; if it's positive, it simply returns the number. We're going to call our function **absolute()**.

Enter the code in **absolute.py**. When the function hits either of the **return** statements, it returns the value of the number (either on its own or multiplied by -1). It then exits the function.

Run the **absolute.py** code and enter the following in the Shell:

```
absolute(10)
absolute(-10)
```

Our last program listing is a classic known as 'FizzBuzz'; as mentioned on page 23, it will help you to understand if, else, and elif.

You also need to know the modulo operator (**%**) for FizzBuzz. This operator returns the remainder from a division. If you don't know how modulo works, watch this video (**magpi.cc/2h5XNRO**).

Now work through the code in **fizzbuzz.py**.

**Happy_birthday.py**

```
def happy_birthday(name):
    count = 0
    while count < 4:
        if count != 2:
            print("Happy birthday to you")
        else:
            print("Happy birthday dear " + name)
        count += 1
```

**Absolute.py**

```
def absolute(number):
    if number < 0:
        return number * -1
    else:
        return number
```

**Fizzbuzz.py**

```
count = 0
end = 100

while count < end:
    if count % 5 == 0 and count % 3 == 0:
        print("FizzBuzz")
    elif count % 3 == 0:
        print("Fizz")
    elif count % 5 == 0:
        print("Buzz")
    else:
        print(count)

    count += 1
```

## Going further

Here are some resources you will find useful.

**GPIO Zero Essentials – magpi.cc/2bA3ZP7**
This Essentials guide book explains how the GPIO Zero Python module provides access to a bunch of features. These are used to hook up electronics to your Raspberry Pi via the GPIO pins.

**FutureLearn – magpi.cc/2h5Sthf**
The Raspberry Pi Foundation has two new online training courses: Teaching Physical Computing with Raspberry Pi and Python, and Teaching Programming in Primary Schools.

**Learning Python – magpi.cc/2h2opWC**
This tutorial provided by The Raspberry Pi Foundation has files you can download. You download the file, called **intro.py**, using this command in a Terminal: `wget http://goo.gl/0ZDOdX -O intro.py --no-check-certificate`. Open the **intro.py** file in IDLE; all the instructions are in the file.

# Importing Code

**Stand on the shoulders of giants by importing other programmers' code**

## Pygame

If you want to learn more about Pygame, check out *Make Games With Python*, our free Essentials Guide to the Pygame module. **magpi.cc/2h2m0vh**

This being the modern world, you're not supposed to do all the work on your own. Instead, you will often stand on the shoulders of other programmers who have done the groundwork for you.

Your programs can import code created by other people using the **import** statement. This enables you to import modules and use their functions – only they're now known as 'methods'.

You import the module at the command line, and then access the functions using dot notation. This is where you list the module, followed by a dot (**.**), then the method.

A common module to use is **math**. This allows you to access lots of maths methods. Open a Python Shell and enter:

```
import math
```

You now have access to all the methods in **math**. You won't notice any difference, but if you type:

```
type(math)
```

...it will say '<class 'module'>'. Let's try out dot notation now. Type **math** followed by a dot and the name of the method (function) you want to use:

```
math.sqrt(16)
```

This gives the square root of 16, which is 4.

Some methods have more than one argument. The **math.pow()** method raises a number to an exponent:

```
math.pow(64,3)
```

This returns 262144.0.

You can also access constant values from a module, which are fixed variables contained in the module. These are like functions/methods, but without the parentheses.

```
math.pi
```

This returns pi to 15 decimal spaces: 3.141592653589793.

```
math.e
```

This returns Euler's number to 15 decimal spaces: 2.718281828459045.

It's also possible to import methods and constants from modules using **from**. This enables you to use them inside your programs without dot notation (like regular functions). For example:

```
from math import pi
from math import e
from math import pow
```

Now, whenever you type **pi** or **e**, you'll get pi and Euler's number. You can also use **pow()** just like a regular function. You can change the name of the function as you import it with **as**:

```
from math import pi as p
```

Now when you enter **p** you'll get pi to 15 decimal spaces. Don't go crazy renaming functions with **as**, but it's common to see some methods and constants imported as single letters.

By creating your own functions, and importing those created by other people in modules, you can vastly improve the capabilities of your programs.

We're going to take everything we've learnt and use it to create a game of Pong; this is one of the world's first videogames.

Write out the code carefully in **pong.py**. Here you'll find variables, functions, loops, and conditional branching: all the stuff we've talked about. Hopefully, you'll now be able to decipher most of this code.

If you're interested in taking Pong further, this program is similar to a version of a Pygame program by Trever Appleton (**magpi.cc/2hgkOUX**). His version has a scorecard and more advanced code. We've kept ours simple so it's easier to start with.

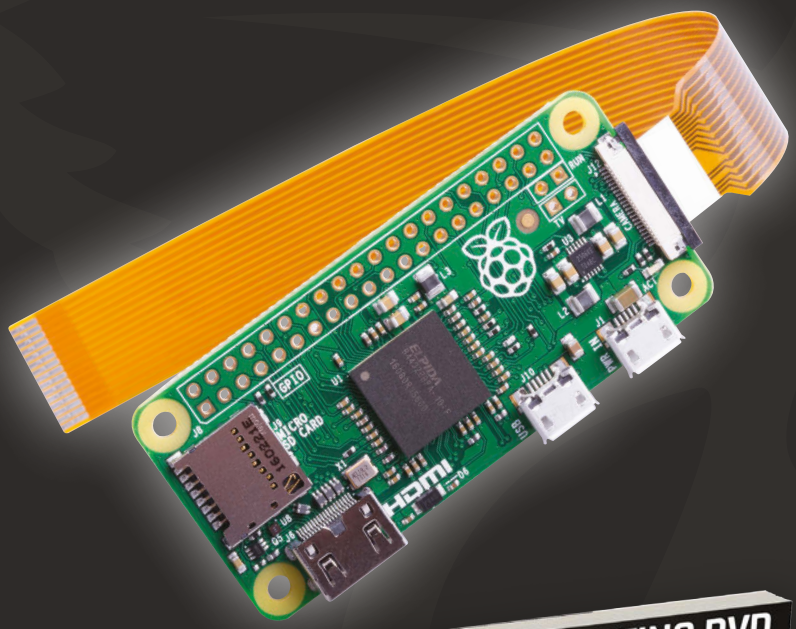
Hopefully this isn't the end of your Python, or programming, journey. There are lots of places you can learn programming from. And we'll have more programming resources for you in every issue of *The MagPi*.

## Pong.py

```
import pygame, sys
from pygame.locals import *

# Set up game variables
window_width = 400
window_height = 300
line_thickness = 10
paddle_size = 50 # try making this smaller for a harder game
paddle_offset = 20

# Set up colour variables
black = (0  ,0  ,0  ) # variables inside brackets are 'tuples'
white = (255,255,255) # tuples are like lists but the values don't change

# Ball variables (x, y Cartesian coordinates)
# Start position middle of horizontal and vertical arena
ballX = window_width/2 - line_thickness/2
ballY = window_height/2 - line_thickness/2

# Variables to track ball direction
ballDirX = -1 ## -1 = left 1 = right
ballDirY = -1 ## -1 = up 1 = down

# Starting position in middle of game arena
playerOnePosition = (window_height - paddle_size) /2
playerTwoPosition = (window_height - paddle_size) /2

# Create rectangles for ball and paddles
paddle1 = pygame.Rect(paddle_offset,playerOnePosition, line_thickness,paddle_size)
paddle2 = pygame.Rect(window_width - paddle_offset - line_thickness, playerTwoPosition, line_thickness,paddle_size)
ball = pygame.Rect(ballX, ballY, line_thickness, line_thickness)

# Function to draw the arena
def drawArena():
    screen.fill((0,0,0))
    # Draw outline of arena
    pygame.draw.rect(screen, white, ((0,0),(window_width,window_height)), line_thickness*2)
    # Draw centre line
    pygame.draw.line(screen, white, ((int(window_width/2)),0),((int(window_width/2)),window_height), (int(line_thickness/4)))

# Function to draw the paddles
def drawPaddle(paddle):
    # Stop the paddle moving too low
    if paddle.bottom > window_height - line_thickness:
        paddle.bottom = window_height- line_thickness
    # Stop the paddle moving too high
    elif paddle.top < line_thickness:
        paddle.top = line_thickness
    # Draws paddle
    pygame.draw.rect(screen, white, paddle)

# Function to draw the ball
def drawBall(ball):
    pygame.draw.rect(screen, white, ball)

# Function to move the ball
def moveBall(ball, ballDirX, ballDirY):
    ball.x += ballDirX
    ball.y += ballDirY
    return ball # returns new position

# Function checks for collision with wall and changes ball direction
def checkEdgeCollision(ball, ballDirX, ballDirY):
    if ball.top == (line_thickness) or ball.bottom == (window_height - line_thickness):
        ballDirY = ballDirY * -1
    if ball.left == (line_thickness) or ball.right == (window_width - line_thickness):
        ballDirX = ballDirX * -1
    return ballDirX, ballDirY # return new direction

# Function checks if ball has hit paddle
def checkHitBall(ball, paddle1, paddle2, ballDirX):
    if ballDirX == -1 and paddle1.right == ball.left and paddle1.top < ball.top and paddle1.bottom > ball.bottom:
        return -1 # return new direction (right)
    elif ballDirX == 1 and paddle2.left == ball.right and paddle2.top < ball.top and paddle2.bottom > ball.bottom:
        return -1 # return new direction (right)
    else:
        return 1 # return new direction (left)

# Function for AI of computer player
def artificialIntelligence(ball, ballDirX, paddle2):
    # Ball is moving away from paddle, move bat to centre
    if ballDirX == -1:
        if paddle2.centery < (window_height/2):
            paddle2.y += 1
        elif paddle2.centery > (window_height/2):
            paddle2.y -= 1
    # Ball moving towards bat, track its movement
    elif ballDirX == 1:
        if paddle2.centery < ball.centery:
            paddle2.y += 1
        else:
            paddle2.y -=1
    return paddle2

# Initialise the window
screen = pygame.display.set_mode((window_width,window_height))
pygame.display.set_caption('Pong') # Displays in the window

# Draw the arena and paddles
drawArena()
drawPaddle(paddle1)
drawPaddle(paddle2)
drawBall(ball)

# Make cursor invisible
pygame.mouse.set_visible(0)

# Main game runs in this loop
while True: # infinite loop. Press Ctrl-C to quit game
    for event in pygame.event.get():
        if event.type == QUIT:
            pygame.quit()
            sys.exit()
        # Mouse movement
        elif event.type == MOUSEMOTION:
            mousex, mousey = event.pos
            paddle1.y = mousey

    drawArena()
    drawPaddle(paddle1)
    drawPaddle(paddle2)
    drawBall(ball)

    ball = moveBall(ball, ballDirX, ballDirY)
    ballDirX, ballDirY = checkEdgeCollision(ball, ballDirX, ballDirY)
    ballDirX = ballDirX * checkHitBall(ball, paddle1, paddle2, ballDirX)
    paddle2 = artificialIntelligence (ball, ballDirX, paddle2)
    pygame.display.update()

```

# FREE PI ZERO!

Subscribe in print for six or 12 months to receive this stunning free gift



## Subscribe today and receive:

- A free Pi Zero v1.3 (the latest model)
- A free Camera Module connector
- A free USB and HDMI cable bundle

**Delivered with your first issue!**

## Other benefits:

- Save up to 25% on the price
- Free delivery to your door
- Exclusive Pi offers & discounts
- Get every issue first (before stores)

FREE DEBIAN + PIXEL LIVE BOOTING DVD
The MagPi
The official Raspberry Pi magazine
BEGINNER'S GUIDE TO CODING
Discover the joy of programming with the biggest and best introductory guide for the Raspberry Pi
THE ONLY PI MAGAZINE WRITTEN BY THE RASPBERRY PI COMMU
YOUR OFFICIAL RASPBERRY PI MAGAZINE
The MagPi
MERRY CHRISTMAS MAKES
AMIGA POWER!
BEGINNER ELECTRONICS
BUILD A DIY ALARM CLOCK
GIVE YOUR ROBOT BRAINS
THE ONLY PI MAGAZINE WRITTEN BY THE RASPBERRY PI COMMUNITY
YOUR OFFICIAL RASPBERRY PI MAGAZINE
The MagPi
BUILD A REMOTE CONTROL ROBOT

# Pricing

## Get six issues:

£30 (UK)
£45 (EU)
$69 (USA)
£50 (Rest of World)

FREE PI ZERO!

## Subscribe for a year:

£55 (UK)
£80 (EU)
$129 (USA)
£90 (Rest of World)

FREE PI ZERO!

## Get three issues:

£12.99 (UK) (Direct Debit)
$37.50 (US) (quarterly)

## How to subscribe:

- **magpi.cc/Subs1** (UK / ROW)
- **imsnews.com/magpi** (USA)
- Call +44(0)1202 586848 (UK/ROW)
- Call 800 428 3003 (USA)

Available on the App Store

GET IT ON Google Play

# SUBSCRIPTION FORM

**YES! I'd like to subscribe to The MagPi magazine and save money**

This subscription is: ☐ For me ☐ A gift for someone*

Mag#53

**YOUR DETAILS** Mr ☐ Mrs ☐ Miss ☐ Ms ☐

First name ..... Surname .....

Address .....

.....

Postcode ..... Email .....

Daytime phone ..... Mobile .....

**If giving* The MagPi *as a gift, please complete both your own details (above) and the recipient's (below).*

**GIFT RECIPIENT'S DETAILS ONLY** Mr ☐ Mrs ☐ Miss ☐ Ms ☐

First name ..... Surname .....

Address .....

Postcode ..... Email .....

**PAYMENT OPTIONS**

**1 DIRECT DEBIT PAYMENT** £12.99 every 3 issues (UK only)
**Instruction to your bank or building society to pay by Direct Debit**

DIRECT Debit

Please fill in the form and send to:
The MagPi, Select Publisher Services Ltd,
PO Box 6337, Bournemouth BH1 9EH

Service user number 8 3 8 7 7 3

Name and full postal address of your bank or building society:

To: The Manager Bank/building society .....

Address .....

.....

..... Postcode .....

Name(s) of account holder(s) .....

Branch sort code ☐☐☐☐☐☐ Account number ☐☐☐☐☐☐☐☐

Reference ☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐ (Official use only)

**Instruction to your bank or building society**
Please pay Select Publisher Services Ltd Direct Debits from the account detailed in this instruction subject to the safeguards assured by the Direct Debit Guarantee. I understand that this instruction may remain with Select Publisher Services Ltd and, if so, details will be passed electronically to my bank/building society.

Signature ..... Date ☐☐/☐☐/☐☐

Banks and building societies may not accept Direct Debit instructions for some types of account.

**SUBSCRIPTION PRICING WHEN PAYING BY CHEQUE OR CREDIT/DEBIT CARD**

6 ISSUES ☐ UK £30 ☐ Europe £45 ☐ Rest of World £50

12 ISSUES ☐ UK £55 ☐ Europe £80 ☐ Rest of World £90

**2 CHEQUE**

I enclose a cheque for ..... (made payable to Select Publisher Services Ltd)

**3 CREDIT/DEBIT CARD** ☐ Visa ☐ MasterCard ☐ Maestro ☐ Switch

Card number ☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐

Expiry date ☐☐ ☐☐ Valid from ☐☐ ☐☐ (if shown)

Issue number ☐☐ (if shown) Security number ☐☐☐
(last 3 digits on the back of the card)

Signature ..... Date ☐☐/☐☐/☐☐

I would like my subscription to begin from issue ..... (month + year)

**RETURN THIS FORM TO:**
*MagPi Magazine* Subscriptions, Select Publisher Services Ltd, PO Box 6337, Bournemouth BH1 9EH

☐ Please tick this box if you DO NOT want to receive any other information from Select Publisher Services Ltd.

☐ Please tick this box if you DO NOT want to receive any other information from other companies.

☐ Please tick this box if you DO NOT want to subscribe to The MagPi newsletter.

**MATT LONG & MARK NICHOLS**

Matt Long and Mark Nichols (pictured) are Microsoft cloud solution architects who work on the Pegasus Mission.
**pegasusmission.com**

# PEGASUS AND THE NORTH AMERICAN EAGLE

## Quick Facts

- The car is 17 metres long
- It has 42,500 horsepower in full afterburner
- The wheels are milled from solid aluminium
- The wheels can endure speeds of over 800mph
- It consumes up to 90 gallons per minute during a run

This land speed record-attempting jet car has a Raspberry Pi sitting right behind the driver

Inside the cockpit of one of the most remarkable machines on Earth sits a Raspberry Pi.

The North American Eagle is an ongoing attempt by an American team to reclaim the land speed record.

The current world record stands at 763.035mph. This was set in October 1997 by British Royal Air Force pilot Andy Green, driving his jet-powered Thrust SSC car.

"We have a mission to break the World Land Speed Record of 763mph (1227.93km/h)," says Brandyn Bayes, team member. "Starting with the battered hulk of a Cold War-era jet plane, we have developed one of the most sophisticated racing machines ever."

**Right** Sensor data from inside the cockpit is gathered by a Raspberry Pi

The North American Eagle started life as a Lockheed F-104 Starfighter. "It was used as a chase plane," reveals Brandyn. These are planes used to follow experimental aircraft and measure engineering data. It followed some of the greatest experimental aircraft in history, among them the X-15 and the Northrop-Grumman B-2 Stealth Bomber.

The particular F-104 that has been transformed into a superfast car was designated 56-0763. By 1998 it was "beyond a shambles," says Brandyn. "The North American Eagle project gave it new life, so that it could once again be one of the fastest machines on Earth."

The Raspberry Pi is placed inside the cockpit, right behind the driver's head
Aluminium wheels are attached to the jet, turning it into a car
The North American Eagle is a Lockheed F-104 jet with its wings removed

## High-speed Pi

While North American Eagle is gunning for the speed record, the Pegasus Mission behind it usually has its eyes to the sky.

"Pegasus Mission uses a high-altitude balloon as a delivery system to carry a payload," reveals Matt Long, a Microsoft cloud solution architect. The balloon is "packed with meteorological sensors to stream telemetry and control flight operations in real time."

It's this expertise in real-time communication that attracted Matt and Mark Nichols (also a Microsoft cloud solutions architect) to the North American Eagle project. "We asked to join the North American Eagle team," Mark tells us.

The Pegasus Mission balloon "provides a rich and interesting experience for users viewing the flight as it happens," Mark explains. "We started with the idea that the Pegasus team could provide a real-time view of what was happening with the vehicle to a global audience."

> "The North American Eagle started life as a Lockheed F-104 Starfighter"

"For this project I built a custom device that was installed in the cockpit of the vehicle," reveals Mark. That device was a Raspberry Pi 3 running Windows 10 IoT Core. The software was written in C# as a Universal Windows Platform application.

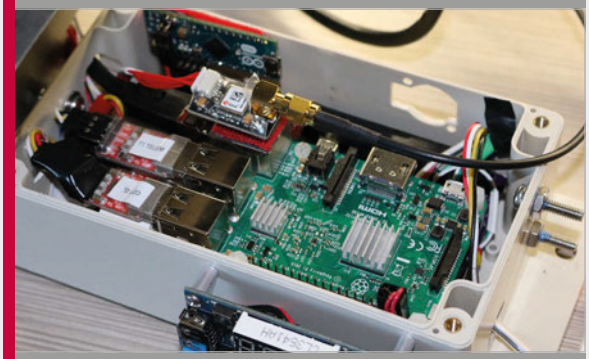
"Microsoft has been a partner supporting North American Eagle for quite a while," says Mark. "Volunteers got together to build this hardware, software, cloud, mobile, and web app capability."

The main aim of the Raspberry Pi was to analyse the car during its test run. "It gathered real-time sensor data," says Mark. GPS, atmospheric, acceleration, and sound level data was gathered by the Raspberry Pi and sent to the Microsoft cloud.

The Pegasus team could "evaluate the benefits of real-time data reporting and analysis in combination with the goals of the North American Eagle team," he explains.

# HIGH-SPEED TELEMETRY

## >STEP-01
### The device
A Raspberry Pi forms the heart of the telemetry device. It's packed with sensors and placed inside the cockpit of the North American Eagle.



## >STEP-02
### Azure cloud
The Raspberry Pi connects to Microsoft Azure Cloud during the speed run. It runs Windows IoT Core and uses software written in C# as a Universal Windows Platform application.




## >STEP-03
### Doing a run
Users can join in on the speed run and send messages to the team, which are displayed during the run. Telemetric data from the run is sent in real-time back to the team on the sidelines.

The more entertaining aspect was the real-time interaction with observers during the test run. "People using [our mobile] apps could also send goodwill messages through the cloud and out to the car, where they were shown on an LCD display," says Mark. The messages were displayed in a screen behind the driver.

Mark also placed a GoPro camera behind the driver to record the user messages on the screen. "The driver could not see them while driving," he explains, "as that would be a distraction, but the camera could see the display and the cockpit."

Naturally, the team needed to filter out inappropriate messages. "The system is configured to check a user message for profanity by calling a third-party SaaS service, Web Purify," says Matt. "If the message passes, then it's sent back into the system and onto the device in the cockpit for display.

"The extra benefit [of the camera] is that it also gave us a front view out of the cockpit," continues Matt. This video was used during a debug session with Jessi Combs (one of the test drivers) after her first run.

"We also invested in two drones to record video, which gave us a great field of view during runs, as well as recording other aspects of the event and its remote location," adds Matt.

## Technology matters

"The technology that makes this real-time and scalable is Azure," Matt reveals.

"Azure provides a delivery mechanism for global communications in real time," he explains. "We can capture the telemetry from the device, messages for users, and analyse or even replay data exactly as it occurred."

Matt tells us that North American Eagle's comprehensive

on-board telemetry can be uploaded to Azure and viewed as a time-series event. "A team of around 20 Microsoft volunteers built mobile applications on Xamarin for iOS, Android, and Windows mobile, and a website in Azure App Services."

"Since day one, people have said this project was just a little crazy," comments Brandyn. "Perhaps that's a bit of an understatement, yet we couldn't agree more. It's that very adventuring spirit that has pushed us to continue to go further, to do something few would ever dare."

Microsoft Research enabled the team to stream live telemetry to fans as the vehicle raced across the empty desert. "We were able to share over 500 million points of telemetry to over 4,500 people on Jessi's first run alone," reveals Brandyn.

"We received messages from users in Australia, New Zealand, UK, France, Norway, and the USA. One of the users watching Jessi Combs's first run sent a message to the device with her approximate top speed before she stopped.

"While we didn't break the records we had hoped for," says Brandyn, "what we found was a reminder of why we love racing: the undeniable rush of adrenaline when things go right, despite the setbacks along the way."

"The North American Eagle team, Ed Shadle, and Jessi Combs have pushed themselves to the limit in the race for a land speed record," adds Matt. "In the process, they have made us all proud members of the human race."

**Above** The North American Eagle uses its jet engine to achieve astonishing speeds

**CLODAGH O'MAHONY**

Masters student and newly appointed teacher of Creative and Innovative Design Technology, Clodagh is also a master rower and archer.
**clodaghomahony.com**

All components are located at the back of the dress
Different colours represent touch to different body parts, recording this data to the Raspberry Pi
The aim was for a dress that would still be fashionable and comfortable, despite the additional tech

# QBEE A SPECULATIVE SOCIAL MEDIA PLATFORM

## Clodagh O'Mahony's university thesis project records touch and voice data to award points for social interaction

### Quick Facts

- Clodagh's Instagram documented the entire process
- The dress was redesigned to improve comfort and functionality
- The Raspberry Pi records touch and keywords
- It then uploads this data to its SQL database
- Conductive thread was used to create 'sectors'

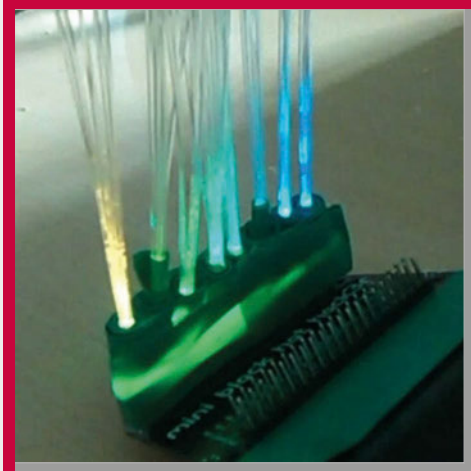
On her website, Clodagh O'Mahony describes herself as a "multi-disciplinary designer with experience in product, graphic, and UX/UI design, as well as illustration and media production."

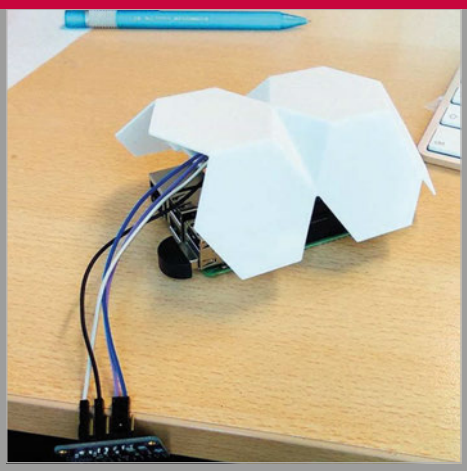
Having completed her BSc in Product Design and Technology at the University of Limerick, Clodagh went on to study for her master's degree at the same establishment, this time in Interactive Media. This is where the Raspberry Pi comes in.

For her thesis project, Clodagh created a dress and an accompanying website to comment on the progression of social media interaction – the idea that it's getting harder and harder to 'hide' on platforms such as Facebook and Twitter due to the sheer amount of personal information we pump into our timelines. Whereas a person could once create an entirely new persona through the predominantly text-based interaction of blogs and chat rooms, we now live a more visual existence online. Photo, video, and emojis have replaced textual communication, adding more 'face' to the name, and inevitably adding more reality. With this in mind, Clodagh set out to design "a wearable connected platform that introduces what is sold as a 'purer' form of social media. The quantitative data means users would have to go to extraordinary lengths to misrepresent their lives, thereby making its information more reliable than that of its competitors."

Clodagh created a corporation named 'QBee', an abbreviation of Queen Bee, with the associated honeycomb theme playing a significant part in the look of both the dress and website. This corporation, if given true life, would provide a range of wearable tech – similar to her dress – that would allow for the recording of social interaction data, updating it to the wearer's online QBee account.

# THE GREAT ELECTRONIC SEWING BEE





## >STEP-01
### Lights
The fibre optics attach to the individual RGB LEDs of the Blinkt, allowing colour control of each cluster. This removes the need to wire multiple LED lights through the dress.

## >STEP-02
### Housing
The 3D-printed casing replicates the hexagonal look of a honeycomb, a theme which is consistently represented across the dress and website.

## >STEP-03
### Wearing
The original dress design fits the Raspberry Pi and other components at the back of the wearer, stylishly incorporating form and function.

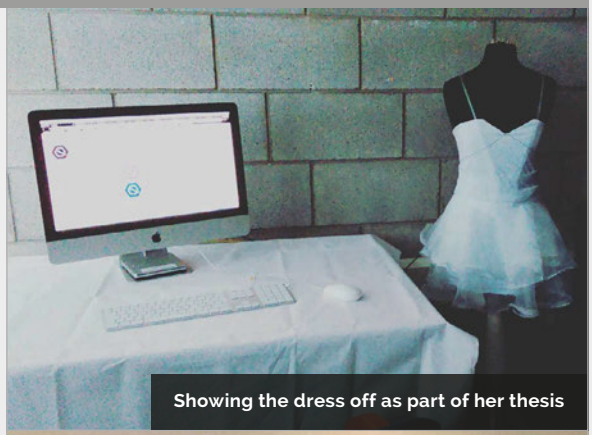
The aim of the build is to record physical interactions between the wearer and the people with whom they come into contact in the real world. A touch to the waist, for example, would be recorded with a certain set of points, whereas a touch to the back would record another. Alongside this physical interaction data, a microphone is used to listen out for any of a series of keywords that are listed as either positive or negative, whereupon the relevant point data can be recorded.

> "A wearable connected platform that introduces what is sold as a 'purer' form of social media"

The build incorporates an Adafruit 12-key capacitive touch sensor breakout board, Pimoroni Blinkt, fibre optics, and a Raspberry Pi, all fitted within a beautiful hexagonal 3D-printed casing.

Clodagh's aim was to use the Blinkt and fibre optics to add colour to the data recording: the touch of a hand to the waist activates the dress to glow a warm purple, a touch to the hip turns it green, and so on.

The dress went through a couple of redesigns throughout the process of the build, all documented on Clodagh's Instagram account (**magpi.cc/2eJgHuZ**), allowing for improvements to cost, comfort, and usability factors. The original dress, though fitting exactly to the design plan of colour-related sectors, wasn't very comfortable. This led Clodagh to create another. Though the second dress doesn't offer exactly the same functionality, it does look the way she wanted, and still uses the Blinkt, though in a slightly different manner. Touch the new dress in any of the sectors and the Blinkt runs through a rainbow sequence until the touch is concluded: it is enough to demonstrate the idea of data recording and capacitive touch.



Showing the dress off as part of her thesis

**Above** Clodagh experimented with multiple 3D-printed cases, finding the ideal location for the tech to be housed

**WENDELL KAPUSTIAK**

A retired investment banker from Venice, Florida, Wendell is an experienced woodworker. He uses a Raspberry Pi to log weather station data and monitor his electricity usage.
**wakrec.blogspot.co.uk**

# SELF-PLAYING PIPE ORGAN

This handcrafted wooden instrument can play any MIDI tune

## Quick Facts

- Wendell first tried making a MIDI wine glass player
- The organ took about three years to build
- He bought the blower on eBay for $80
- Pipes range from 9 to 40 inches in length
- He's now rebuilding the organ in solid oak

Experienced woodworker Wendell Kapustiak needed all his carpentry skills to create this impressive self-playing pipe organ (**magpi.cc/2fMacLy**). "The most fundamental [problem] was that I had no idea how a pipe organ actually worked," admits Wendell, who eventually based its mechanical workings on a project by Matthias Wandel (**woodgears.ca**). The most difficult part was making the 42 wooden pipes, which span three-and-half octaves, since each one has unique dimensions to produce the correct pitch. For this, Wendell used information provided by Raphi Giangiulio's YouTube videos (**youtu.be/-mibK_Dp-ZY**).

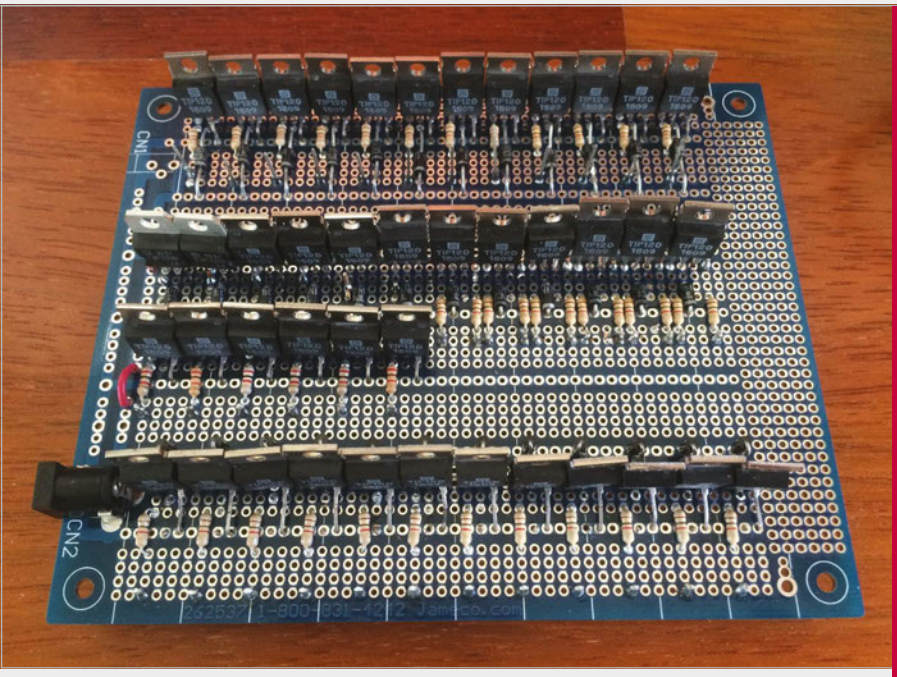
The pipes are linked via PVC plumbing to a wind chest powered by a Kooltronic KBR125 blower, as used in data centres. "I had originally tried a small shop vac as a blower," Wendell tells us. "It had two problems. The first was that it was very noisy; the second was that it ran hot. When I enclosed it in a box to control the noise, it got so hot that my first one burned out."

To make it play, each wooden pipe has a valve opened and closed by a solenoid, triggered from an Arduino Due via a power-boosting driver board. The 'brains' of the operation is a Raspberry Pi, which performs three main functions. As well as a graphical

Each of 42 handcrafted wooden pipes produces a different musical note

When triggered, solenoids open their respective pipe valves to play notes

The solenoids are wired indirectly to an Arduino Due receiving data from a Pi



**Above** A home-built driver board with transistors is used to boost the power from the Arduino output to trigger the solenoids

user interface for selecting music to play, it converts the MIDI binary files into delay/note-on/note-off commands, plus musical directions: "The Raspberry Pi takes into consideration tempo changes and any other subtleties in note timing, and converts them into a number of microseconds," explains Wendell. "In this way, the Pi has done all the heavy lifting as far as calculations go." Another Python program is then used to send this data to the Arduino via USB.

> "The Raspberry Pi takes into consideration tempo changes and any other subtleties"

While the high-tech side of the project proved fairly hassle-free, the physical engineering was more problematic. One difficulty was the wind pressure regulator, which originally had a bellows-type mechanism. "I was trying out a few different designs, trying to develop one that was mechanically simple and dependable while producing a very stable pressure." Fortunately, Wendell's friend Jim, a retired HVAC engineer, stopped by and suggested a new approach, using a weight-controlled door as used in ventilation systems. "I whipped up a sample in about 20 minutes and it worked a treat."

Another issue involved the spacing of the pipes, which Wendell discovered could interact with each other if placed too closely together. "I suddenly understood why organs were designed with the pipes in a few rows but spread out, not bunched together." He also reversed the overall layout to put the solenoids at the front, so listeners can see them operating the valves. He has since added some LEDs: "It makes it much easier to follow what's going on than just watching the solenoids."

The finished pipe organ generated a lot of interest from attendees at the Orlando Maker Faire. "I think most people were able to walk away with some level of insight as to how the combination of the old technology of the pipe organ and the new technology of the computer and microcontroller fit together."

## MAKING SWEET PIPED MUSIC

### >STEP-01

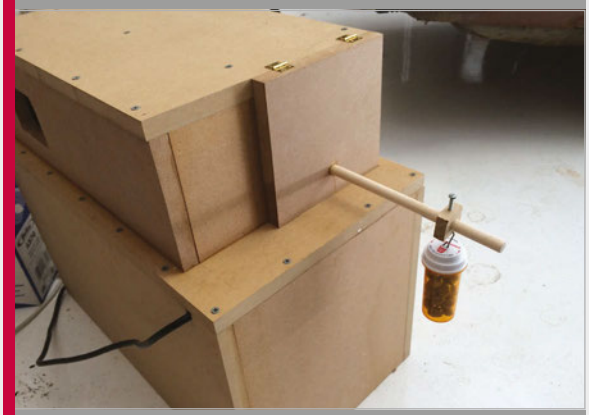
**Wooden pipes**

Each of 42 wooden pipes produces a different note. While larger usually means lower, some of the bottom notes are produced by shorter pipes containing an airtight stopper to reduce the pitch.

### >STEP-02

**Solenoid valves**

Solenoids are used to open and close the pipe valves to play and stop notes. O-rings prevent them from sticking in the on position, while also stopping the plungers making a clacking noise.



### >STEP-03

**Wind chest**

The wind chest features a weighted hinged door which shuts automatically when many pipes are playing, in order to boost the air pressure. The wind is provided by a Kooltronic KBR125 blower.

INITIAL STATE

# PIOT CHALLENGE WINNER!

**BRIAN BRANDAW**

A 25-year veteran of the IT industry, Brian is highly experienced in Linux and enjoys playing around with his Pi at home.

# WATER TANK LEVEL MONITOR

Automated water collection and monitoring through the power of Raspberry Pi – and the winner of Initial State's **PIoT Challenge**

## Quick Facts

- It took four months on-and-off to build
- Brian reckons he could rebuild one in about four hours
- The ProtoZero board allows you to build circuits direct to the GPIO
- It's accessible and can be controlled wirelessly
- Brian has other Pis monitoring parts of his property

The Raspberry Pi is no stranger to automation, and if you've read enough issues of this magazine you'll have seen plenty of home automation projects that use it. This automation project is a little different, though: measuring the water level in a couple of water tanks, a solution devised by Brian Brandaw from Texas.

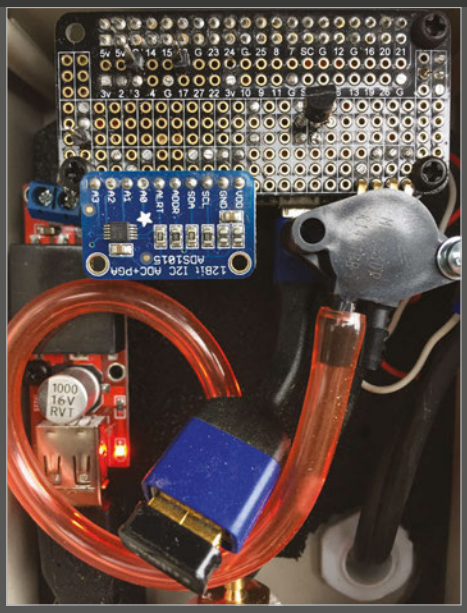
"Being on a rural lot, we rely on a private well for our water," Brian explains. "The quality is quite poor and the well has been unreliable at times. We took advantage of the metal roof on the barn and the existing rain gutters and installed two rainwater collection tanks. We added standard float valves to these to fill water troughs for the horses. This arrangement worked well until we ran out of water in the tanks! There is no easy means to track tank level."

All the traditional methods are visual, requiring you to go out to the tanks to check them on a fairly regular basis, although some can be used at a distance.

"I wanted to devise a means to monitor tank levels to know when things were running low," continues Brian. "We can also measure water consumption by looking at how the water level changes over time."

The system is wired up to a ProtoZero board on the Pi Zero
The pressure sensor gets its reading from the tank
It can be operated wirelessly from another system

# MONITORING WATER



## >STEP-01
### Under pressure
The pressure sensor at the bottom of the tank is constantly under pressure, although this varies as the water level goes up and down.

## >STEP-02
### Light reading
The Pi Zero reads the voltage levels on the two outputs with the help of an analogue-to-digital converter. The GPIO cannot measure it natively.

## >STEP-03
### Simple geometry
The Pi Zero uses fluid dynamics and the measurements of the tank to determine the volume of the water in the tank. This can be charted to keep track of consumption.

As we live in 2016, computer-aided monitoring was on the table. Brian's method was rather ingenious: instead of measuring the water level at certain points, he measured the pressure of the water at the bottom of the tank. With a few extra calculations to take into account atmospheric pressure and the size of the tank, the volume of water could be calculated.

> "I wanted to devise a means to monitor tank levels to know when things were running low"

"I used a Pi Zero as the platform to read the pressure sensor (via an analogue-to-digital converter), calculate the tank volume, and post the data to InitialState for dashboarding and alerting," Brian tells us.

His experience with Linux throughout his career made him more comfortable creating the project, basing it on a similar solution employing an Arduino that he'd seen a few years ago. This way he managed to save money on proprietary solutions as well.

"It was tricky to build, but not because of the complexity of the solution," Brian notes. "The box I chose is rather small, so it took some time to squeeze it all in there. The biggest challenge is the tight space on the ProtoZero board. I'm not the best at soldering, so it's been slow going at times."

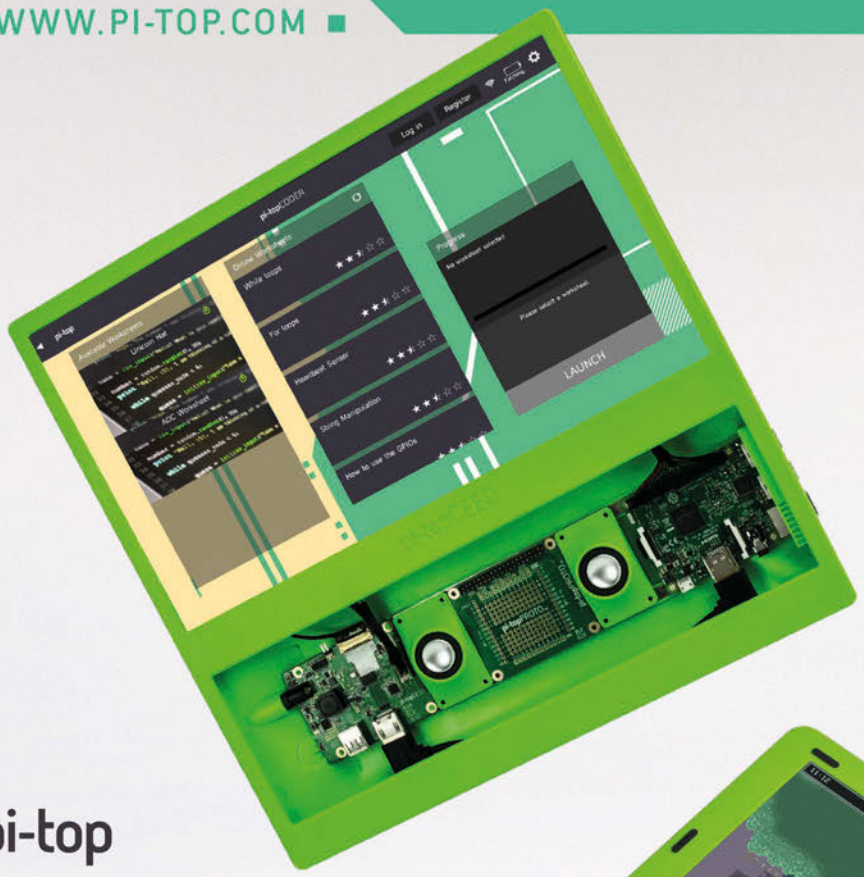
Brian reports that the solution works well, with only a minor repair required after he damaged a connection. In the future he plans to upgrade these connectors to make them a little more rugged, but otherwise it looks like his project has helped out enormously with his water supply.

**Above** The whole thing is connected to the bottom of the tank in a small grey box

# pi-top

LEARN
PLAY
CREATE

■ WWW.PI-TOP.COM ■



## pi-topCEED

- Adjustable Viewing Angles
- 14" HD Screen
- Modular Components

**$114.99**
without Raspberry Pi
ex VAT

## pi-top

- 10 Hour Battery Life
- 13.3" HD Screen
- Modular Components

**$264.99**
without Raspberry Pi
ex VAT

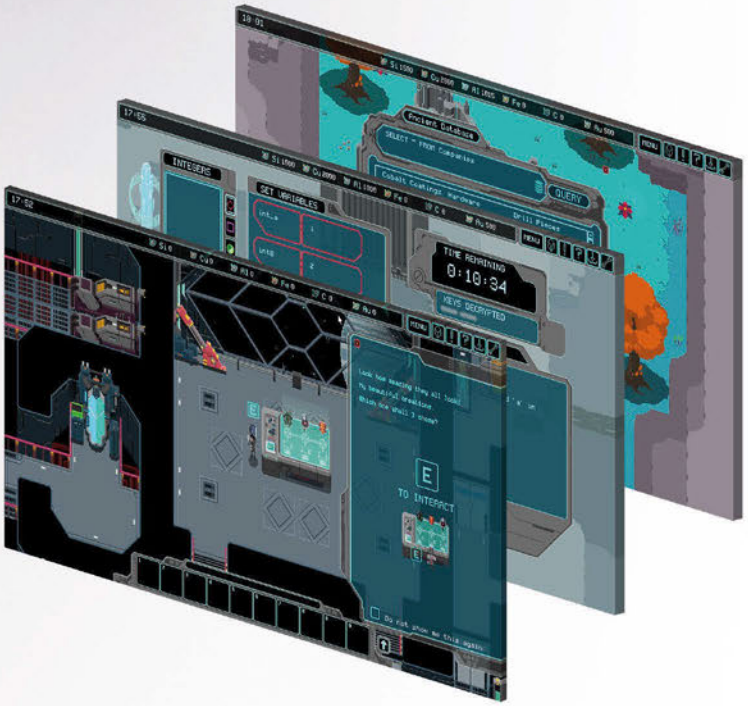
Available in green or grey colours

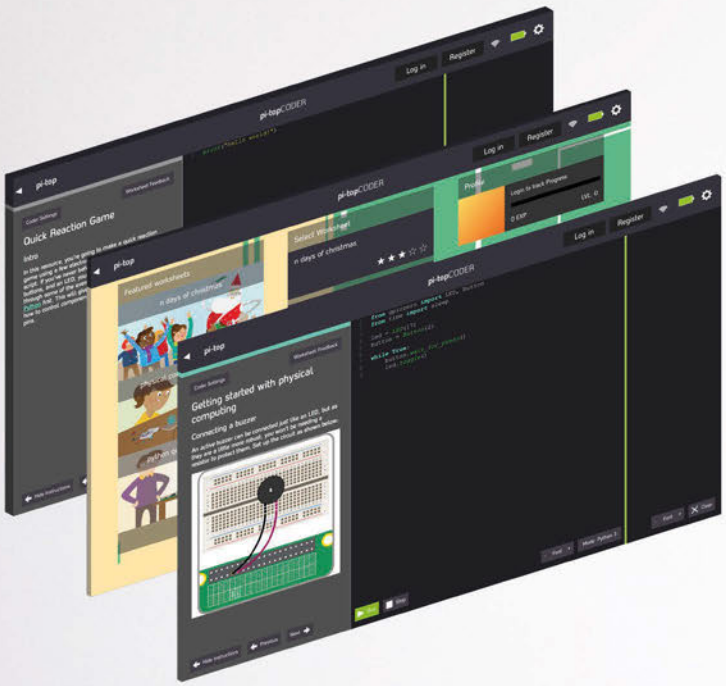
pi-top | Worldwide shipping available in green or grey at www.pi-top.com
Stay up to date with our latest news by following our social media

## CEED**universe**

CEED**universe** is a world of fantasy grounded in computing reality! After crash landing on a strange new planet you will first encounter 'drag and drop' coding puzzles that improve your computational thinking skills.

Discover more about the planet you've landed on and the civilisation that used to inhabit it while learning harder and more complex concepts in fun and engaging ways! Before long you'll be writing your own conditional statements, loops and functions. Find out more about CEED**universe** at www.pi-top.com





## **pi-top**CODER

Influenced by the workflow of makers and hackers, pi-top presents **pi-top**CODER - an integrated code editor which allows you to learn, write and test code all in one view. With intuitive syntax highlighting, dynamic views and customizable themes it makes for a versatile learning tool for your projects.

**pi-top**CODER also has every Raspberry Pi Foundation lesson plan created and will track and save your progress as you go through dozens of fun hardware and software projects.

## **pi-top**PROTO

**pi-top**PROTO is a HAT compatible Add-on Board for your **pi-top** or **pi-top**CEED that allows you to prototype electronics. Create a Weather Station, HAM Radio, Heart Rate Monitor, or integrate any Arduino based maker kits into your own Raspberry Pi compatible prototyping board!

## **pi-top**SPEAKER

Give a voice to your **pi-top** device with **pi-top**SPEAKER!

- Modular design, attach up to three in a row to give true stereo sound.
- 2W per module
- Left, Right and Mono mix selection
- High quality SPDIF digital audio from HDMI
- I²C controlled

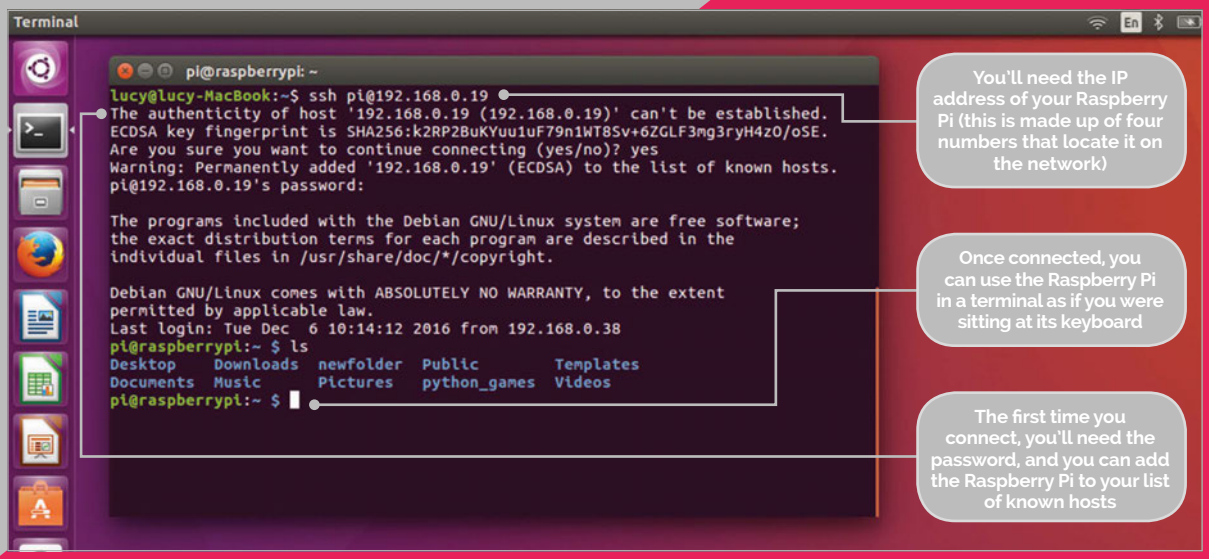
Our Official Distributors
adafruit
HANDS-ON SCIENCE
KIWI electronics
KUBII
MODMYPI
MICRO CENTER computers & electronics
Pi-Shop.ch
RS
sparkfun ELECTRONICS

# BEGINNER'S GUIDE TO SSH

Connect to your Raspberry Pi remotely through a terminal using Secure Shell

## You'll Need

- Raspberry Pi
- Raspbian with PIXEL
- SSH client


Terminal
pi@raspberrypi: ~
lucy@lucy-MacBook:~$ ssh pi@192.168.0.19
The authenticity of host '192.168.0.19 (192.168.0.19)' can't be established.
ECDSA key fingerprint is SHA256:k2RP2BuKYuu1uF79n1WT8Sv+6ZGLF3mg3ryH4zO/oSE.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.168.0.19' (ECDSA) to the list of known hosts.
pi@192.168.0.19's password:
The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue Dec 6 10:14:12 2016 from 192.168.0.38
pi@raspberrypi:~ $ ls
Desktop Downloads newfolder Public Templates
Documents Music Pictures python_games Videos
pi@raspberrypi:~ $
You'll need the IP address of your Raspberry Pi (this is made up of four numbers that locate it on the network)
Once connected, you can use the Raspberry Pi in a terminal as if you were sitting at its keyboard
The first time you connect, you'll need the password, and you can add the Raspberry Pi to your list of known hosts

Secure Shell (better known as 'SSH') is an encrypted networking technology that enables you to manage computers from the command line over a network.

SSH is handy if you want to quickly connect to your Raspberry Pi from a terminal on another computer. It's also ideal for lightweight distro installations that don't have an interface. It's especially useful when creating Internet of Things (IoT) projects, as these may be embedded and not require a desktop.

We've already looked at VNC (Virtual Network Computing), and SSH offers a similar service. But while VNC shares the entire desktop, SSH works from the command line.

On Linux PCs and Macs, you don't need to install any software to start using SSH. Linux and Mac OS X have the SSH command-line application installed by default; you can view its manual in the terminal using `man VNC`.

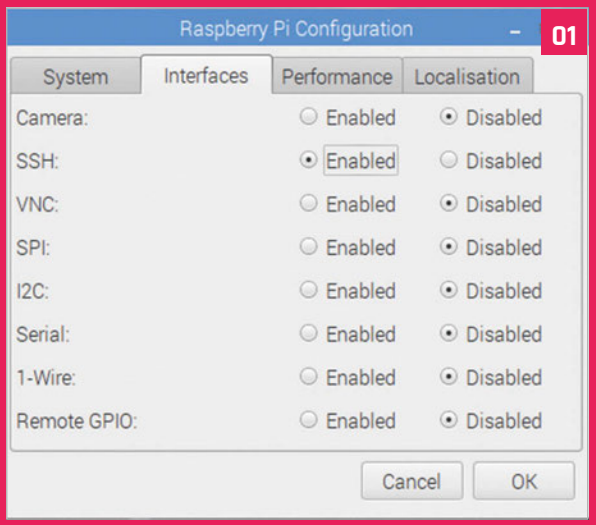
On Windows you will need to download an SSH client; the most commonly used one is called PuTTY. Download the PuTTY software from Simon Tatham's website (**magpi.cc/2hb1IiW**).

You'll need to use the password for your Raspberry Pi to log in using SSH. For security reasons, we recommend changing the default password.

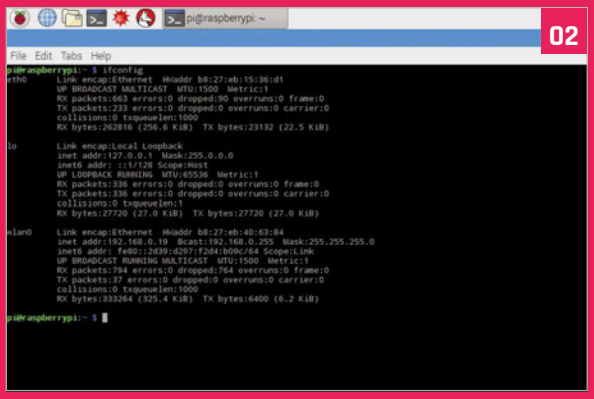
SSH uses an encrypted network, so it doesn't send your password as plain text. More advanced users can control the encryption keys, using ssh-keygen. For now, we'll look at setting up and using SSH.

## >STEP-01
### Activate SSH

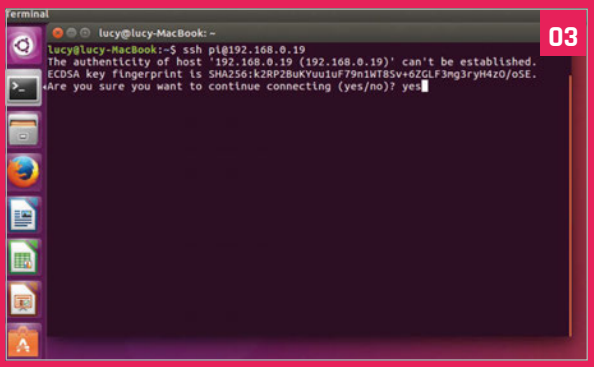
As of the November 2016 release of Raspbian with PIXEL, SSH is no longer turned on by default. On your Raspberry Pi, choose **Menu > Preferences > Raspberry Pi Configuration**. Click on Interfaces and set SSH to Enabled. Click OK.



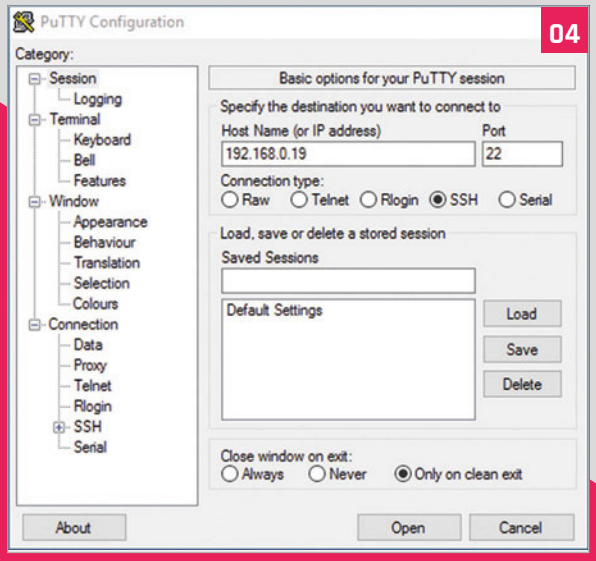
## >STEP-02
### Get the IP address
Connect your Raspberry Pi to a local network. Use a wireless network, or connect the Raspberry Pi directly to a router with an Ethernet cable. Open a terminal and enter **ifconfig** to find the IP address. With Ethernet, it'll be the four numbers next to **inet addr:**, such as 192.168.0.27. If you're connected wirelessly, look for similar numbers under **wlan0**.



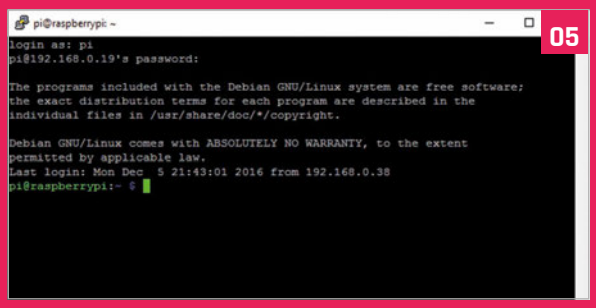

## >STEP-03
### SSH
On a Linux or Mac, open a terminal and enter **ssh pi@youripaddress**. On our network, that's **ssh pi@192.168.0.19**. The first time, you'll get this message: 'The authenticity of host (192.168.0.19') can't be established. ECDSA key fingerprint is SHA256:' followed by a long cryptographic hash of letters and numbers. It will say 'Are you sure you want to continue connecting?'. Enter **yes** and press **RETURN**.


03
lucy@lucy-MacBook: ~
lucy@lucy-MacBook:~$ ssh pi@192.168.0.19
The authenticity of host '192.168.0.19 (192.168.0.19)' can't be established.
ECDSA key fingerprint is SHA256:k2RP2BuKYuu1uF79n1WT8Sv+6ZGLF3ng3ryH4zO/oSE.
Are you sure you want to continue connecting (yes/no)? yes

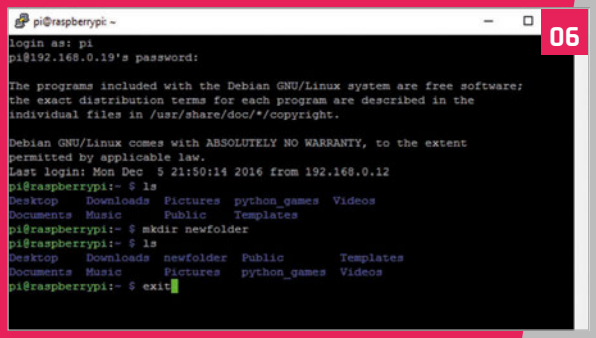
## >STEP-04
### PuTTY
On a PC you'll need to install PuTTY. Download the **putty.exe** file and click Run. The PuTTY Configuration window appears with basic options. Enter the IP address of your Raspberry Pi in the 'Host Name (Or IP Address)' field. Don't change the 'Port' field. Click Open. You will get a PuTTY 'Security Alert' field. Click Yes. The terminal window displays 'login as:' Enter **pi** and press **RETURN**. Now enter the password for your Raspberry Pi.




## >STEP-05
### The command line
You will now see your usual command line replaced with **pi@raspberrypi: ~$**. You are now logged in and working on the command line from your Raspberry Pi. Enter **ls** and you'll see **python_games** along with the other unique Raspberry Pi folders and files. You can create, edit, move, and work with files as if you were using a terminal on your Raspberry Pi.


05
pi@raspberrypi: ~
login as: pi
pi@192.168.0.19's password:
The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Mon Dec 5 21:43:01 2016 from 192.168.0.38
pi@raspberrypi:~ $

## >STEP-06
### Exiting
There are limitations over VNC. You can't open programs with a graphical interface, so you'll need to use command-line alternatives (such as nano or vim instead of Leafpad for text editing). It's not as easy to share files using SSH as it is with VNC, but for fast command-line editing, it's hard to beat. Enter **exit** at the command line to finish.


06
pi@raspberrypi: ~
login as: pi
pi@192.168.0.19's password:
The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Mon Dec 5 21:50:14 2016 from 192.168.0.12
pi@raspberrypi:~ $ ls
Desktop Downloads Pictures python_games Videos
Documents Music Public Templates
pi@raspberrypi:~ $ mkdir newfolder
pi@raspberrypi:~ $ ls
Desktop Downloads newfolder Public Templates
Documents Music Pictures python_games Videos
pi@raspberrypi:~ $ exit

# GET STARTED WITH THE PI CAMERA

Snap photos from your Raspberry Pi using its special, programmable camera

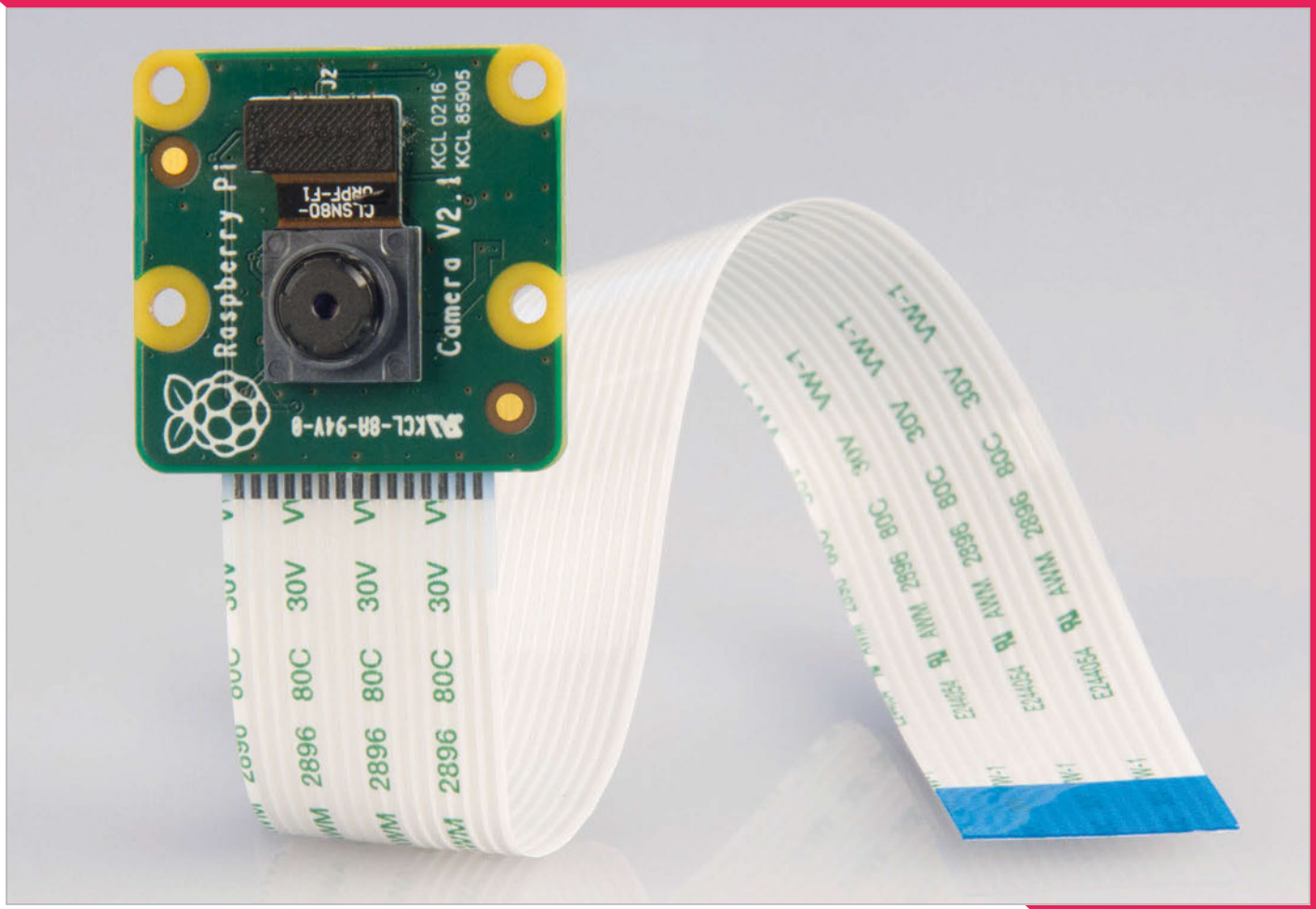
## You'll Need

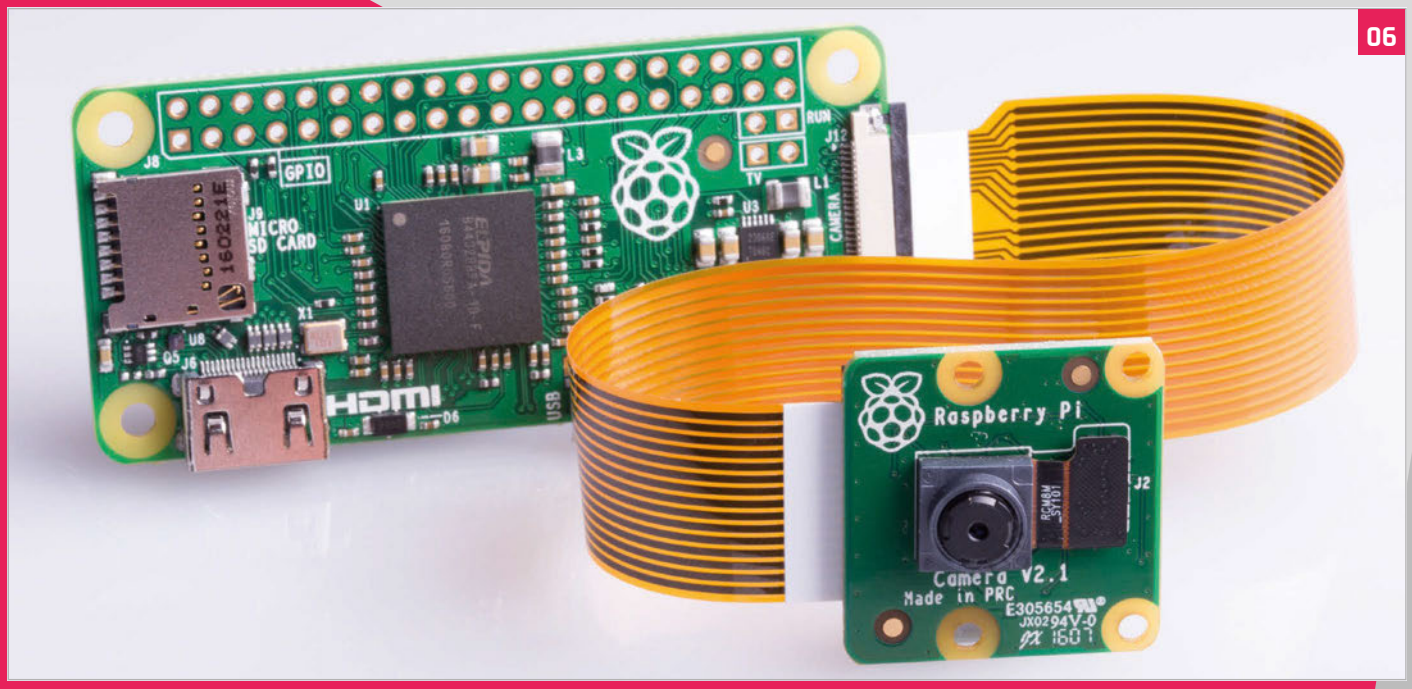
- Raspberry Pi
- Raspbian
- Pi Camera Module
  magpi.cc/28ljlsz

The Raspberry Pi has a few mysterious connectors on it that you wouldn't normally use when hooking it up. We've covered the GPIO pins in a previous issue, but now we're going to move onto the CSI port. You'll find this located between the HDMI and audio jack on a normal Pi, and on the edge of a Pi Zero. CSI stands for Camera Serial Interface and, as the name suggests, it's used to connect a camera to the Raspberry Pi. Not just any camera either: specifically, the Raspberry Pi Camera Module.

The Camera Module, so called because it looks like a piece of circuit board and is attached via a ribbon cable, is a special programmable camera for the Raspberry Pi. It can take photos and video, and has many extra functions like time-lapse photography and slow-motion recording. It's fairly easy to control from the command line, or by using specific code in a Python script.

With the recent addition of a camera port to the Pi Zero, every Raspberry Pi can use the camera for some cool and fun projects. Here's how to get started with it.



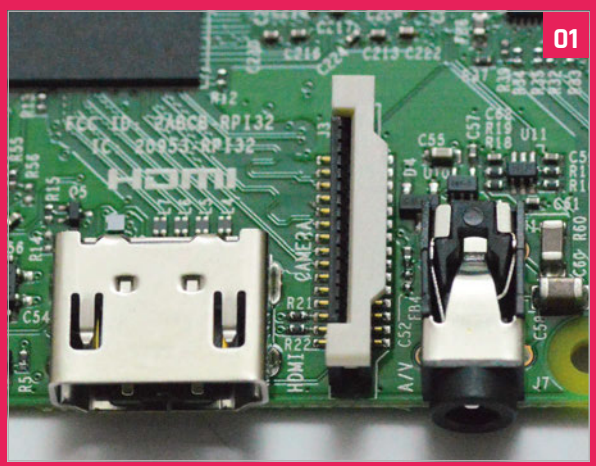


06

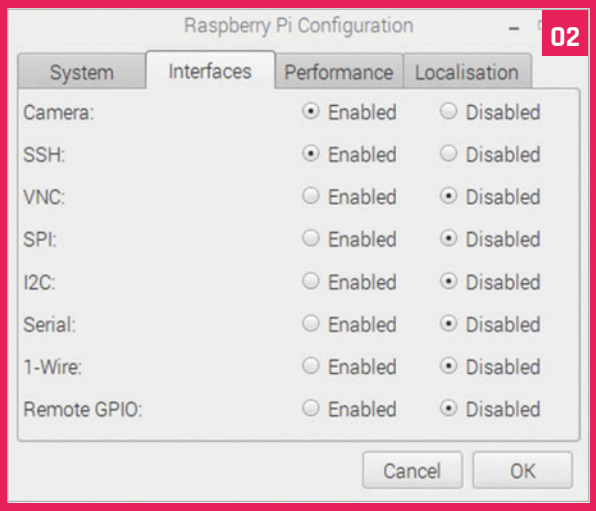
## >STEP-01
### Attach the camera
With the Pi turned off, gently lift up the plastic catch of the CSI connector. Take the end of the ribbon and insert it into the slot, with the silver connectors facing towards the HDMI port, before pushing the catch firmly back down.

## >STEP-02
### Enable the camera
Boot up the Raspberry Pi. Once you're in the desktop environment, head to **Menu > Preferences** and select Raspberry Pi Configuration. Go to the Interfaces tab and click the option to enable the camera.

## >STEP-03
### Taking a photo
You can take a photo from the terminal by typing `raspistill -o image.jpg`. This will show a preview for five seconds, then shoot an image at maximum resolution and save it as **image.jpg** by default.



01



02

## >STEP-04
### Recording video
Video is a little more complex: you need to tell it how long the video is in milliseconds. For ten seconds, use `raspivideo -t 10000 -o video.h264`. It'll preview what you're shooting, and the file will be at 720p.

## >STEP-05
### Python programming
Controlling the camera with Python is easy: all you need to do is import the `picamera` module at the beginning of your script. There's some info on how it works here: **magpi.cc/2gSZf9L**.

## >STEP-06
### Pi Zero
The updated Raspberry Pi Zero adds a special connector to attach the camera. You'll need to get an adapter cable to plug it in (**magpi.cc/2gT2KwE**), but otherwise it works the same.

**PHIL KING**

When not sub-editing *The MagPi* and writing articles, Phil loves to work on Pi projects, including wheeled robots.
**@philking68**

# MAKE A PUSH BUTTON MUSIC BOX

Use two or more tactile push buttons to play different sound samples

## You'll Need

- GPIO Zero
- 1× solderless breadboard
- 2× push buttons
- 3× male-to-female jumper wires
- 2× male-to-male jumper wires
- Headphones or speakers

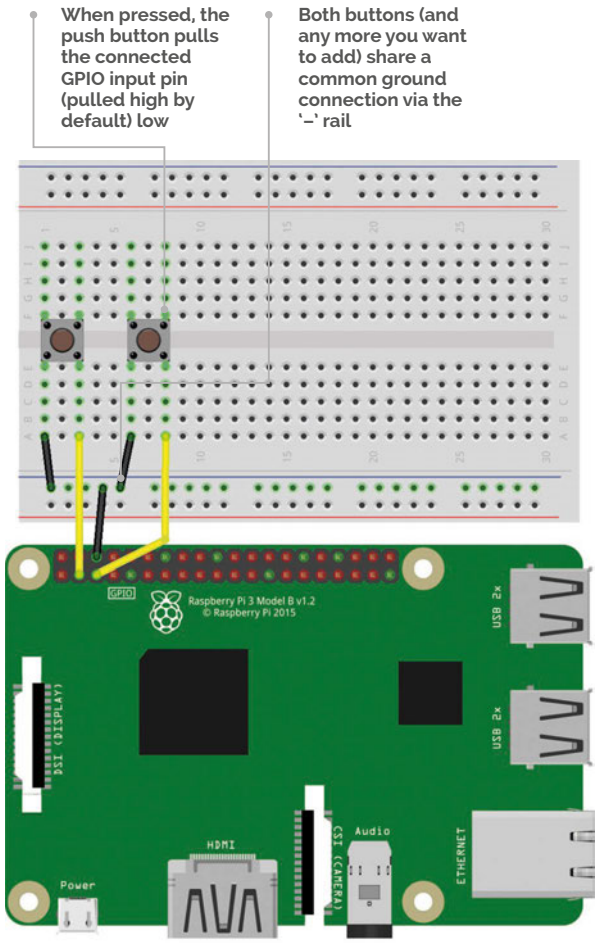
In this easy electronics project, we'll use push buttons to make a GPIO music box that triggers different sounds when we press the buttons.

## >STEP-01

### Get some sounds

Open a Terminal window and create a new folder: **`mkdir musicbox`**. Switch to it with **`cd musicbox`**. Now we need to source some sound samples. While there are many public domain sounds online, we'll use some of Scratch's built-in percussion sounds, already present on the Pi. In your Terminal, enter **`mkdir samples`**, then change to that directory: **`cd samples`**. Now copy the Scratch percussion sounds with:

```
cp /usr/share/scratch/Media/Sounds/
Percussion/* .
```


When pressed, the push button pulls the connected GPIO input pin (pulled high by default) low
Both buttons (and any more you want to add) share a common ground connection via the '–' rail
GPIO
Raspberry Pi 3 Model B v1.2
© Raspberry Pi 2015
USB 2x
USB 2x
DSI (DISPLAY)
HDMI
CSI (CAMERA)
Audio
ETHERNET
Power

## >STEP-02

### Play a drum

First, we'll create a simple Python program to play a drum sample repeatedly. Open Python 3 (IDLE), create a new file, and enter the code from **ch4listing1.py**. Save the file in your **musicbox** folder. Press **F5** to run the program and listen to it play. If you can't hear it, you might need to alter your audio configuration; in a Terminal, enter **`amixer cset numid=3 1`** to switch it to the headphone socket, or **`amixer cset numid=3 1`** to switch to the HDMI output.

## >STEP-03

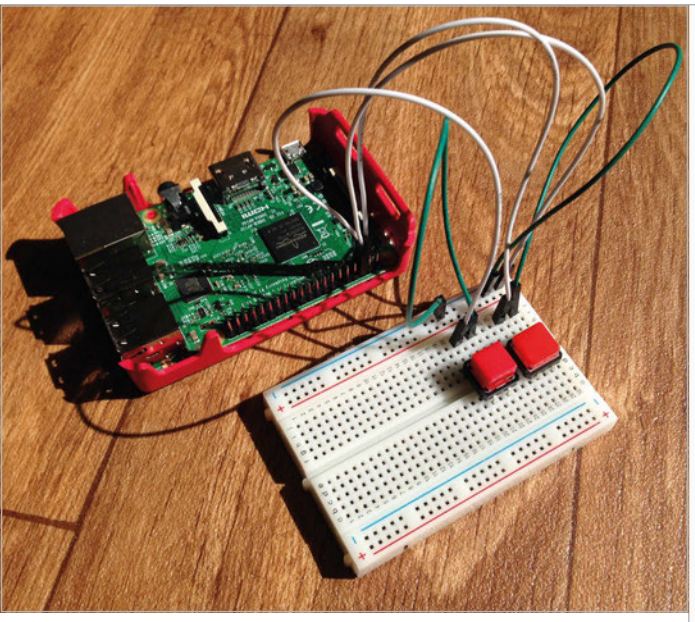
### Wire up a button

Turn off the Pi to start building a circuit. Place a push button so it straddles the central groove of the breadboard. One leg is connected to GPIO pin 2, and the other to the common ground rail on the breadboard, which in turn is wired to a GND pin.

We'll make a sound play when the button is pressed. Open a new file in Python 3 IDLE, enter the code from **ch4listing2.py**, and save it in your **musicbox** folder. At the start, we import the **`Button`** class from GPIO Zero, and the **`pause`** class from the **`signal`** library. We assign the button variable to GPIO pin 2, with **`button = Button (2)`**. We tell the sound to play when the button is pressed, using **`button.when_pressed = drum.play`**. Finally, we add **`pause()`** at the end so that the program will continue to wait for the button to be pressed. Run the program and every time you press the button, the drum sound should play.

Language
>PYTHON 3

DOWNLOAD:
magpi.cc/2bhwqlH



**Below** Each time you press a button, the assigned sound sample will play through a connected speaker

## >STEP-04

### Add a second button

We'll now add a second button, as in the diagram. Wire it to GPIO 3 and the ground rail. Open a new file in Python 3 IDLE, enter the code from **ch4listing3.py**, and save it in **musicbox**. Note that rather than assigning the **Button** objects and sounds to the pins individually, we're using a dictionary structure to assign their numbers to sound samples:

```
sound_pins = {
    2: Sound("samples/DrumBizz.wav"),
    3: Sound("samples/CymbalCrash.wav"),
}
```

We then create a list of buttons on all the pin numbers in the **sound_pins** dictionary: **buttons = [Button(pin) for pin in sound_pins]**

Finally, we create a **for** loop that looks up each button in the dictionary and plays the appropriate sound:

```
for button in buttons:
    sound = sound_pins[button.pin.number]
    button.when_pressed = sound.play
```

Run the program and press each button to hear a different sound.

## >STEP-05

### Add more buttons

The way we have structured the program makes it easy to add extra buttons and assign them to sound samples. Just connect each button to a GPIO number pin (not any other type) and the ground rail, as before. Then add the GPIO pin numbers and sounds to the dictionary, as in the following example:

## ch4listing1.py

```
import pygame.mixer
from pygame.mixer import Sound

pygame.mixer.init()

drum = Sound("samples/DrumBuzz.wav")

while True:
    drum.play()
```

## ch4listing2.py

```
from gpiozero import Button
import pygame.mixer
from pygame.mixer import Sound
from signal import pause

pygame.mixer.init()
button = Button(2)
drum = Sound("samples/DrumBuzz.wav")

button.when_pressed = drum.play
pause()
```

## ch4listing3.py

```
from gpiozero import Button
import pygame.mixer
from pygame.mixer import Sound
from signal import pause

pygame.mixer.init()

sound_pins = {
    2: Sound("samples/DrumBuzz.wav"),
    3: Sound("samples/CymbalCrash.wav"),
}

buttons = [Button(pin) for pin in sound_pins]
for button in buttons:
    sound = sound_pins[button.pin.number]
    button.when_pressed = sound.play

pause()
```

```
sound_pins = {
    2: Sound("samples/DrumBizz.wav"),
    3: Sound("samples/CymbalCrash.wav"),
    4: Sound("samples/Gong.wav"),
    14: Sound("samples/HandClap.wav"),
}
```

**SIMON LONG**
Works for Raspberry Pi as a software engineer, specialising in user interface design. In his spare time he writes apps for the iPhone and solves crosswords.
**raspberrypi.org**

# AN INTRODUCTION TO C PART 07

# ARRAYS AND STRINGS

How to handle arrays (lists of values) and strings (lists of letters) in C

**KEEP INSIDE YOUR ARRAY**

One of the nastiest sources of crashes and bugs in C is creating an array and then writing past the end of it. The compiler won't stop you writing to memory off the end of an array, and doing so can have serious consequences. Always make sure your array indices fit inside your array.

The variables we've looked at so far are all single numeric values. In this instalment, we're going to look at how C handles lists of values, and that leads into using lists of letters to store and manipulate text strings.

An *array* is a variable which stores multiple values of the same type; the individual values are accessed by *indexing* the array. An array can have one or more *dimensions*; a one-dimensional array is a single list of values, while a two-dimensional array is a list of lists of values, and so on.

An array is declared in C by putting the size of each dimension in square brackets after the variable name. So:

```
int a[10];
```

is a list of 10 integers, while

```
int b[5][6];
```

is a list of five lists, each containing six integers.

When accessing the elements inside an array, the array index – the number inside the brackets – starts at 0. So the integers contained within **a** above are referred to as **a[0]**, **a[1]**, **a[2]** and so on, up to **a[9]**.

**Below** Array elements are stored sequentially in memory, with the array name a pointer to the first element. Multi-dimensional array elements are stored with the elements with neighbouring values in the rightmost index next to each other

## Arrays and pointers

The name of an array is effectively a pointer to the first element of the array. An array is a contiguous block of memory which contains all the elements in order, so you can use a pointer to access it.

Here's an example:

```
#include <stdio.h>
void main (void)
{
   int a[10];
   int count;
   for (count = 0; count < 10; count++)
   {
        a[count] = count * 10 + count;
   }
   printf ("The first and second elements
of a are %d and %d\n", a[0], a[1]);
   printf ("Or, as pointers, %d and %d\n",
*a, *(a+1));
}
```

This fills the ten values of **a** with the numbers 0, 11, … and then reads **a[0]** and **a[1]**. It then reads the same values using **a** as a pointer; running the code shows that they are identical.

With a multi-dimensional array, you need to consider how the compiler arranges the dimensions in memory. The elements at the rightmost index of the array are grouped together. With the array **b[5][6]**, **b** itself points at **b[0][0]**. **b+1** points at **b[0][1]**, **b+5** points at **b[0][5]**, and **b+6** points at **b[1][0]**.

You can initialise an array at the same time as you declare it by putting the values in curly brackets:

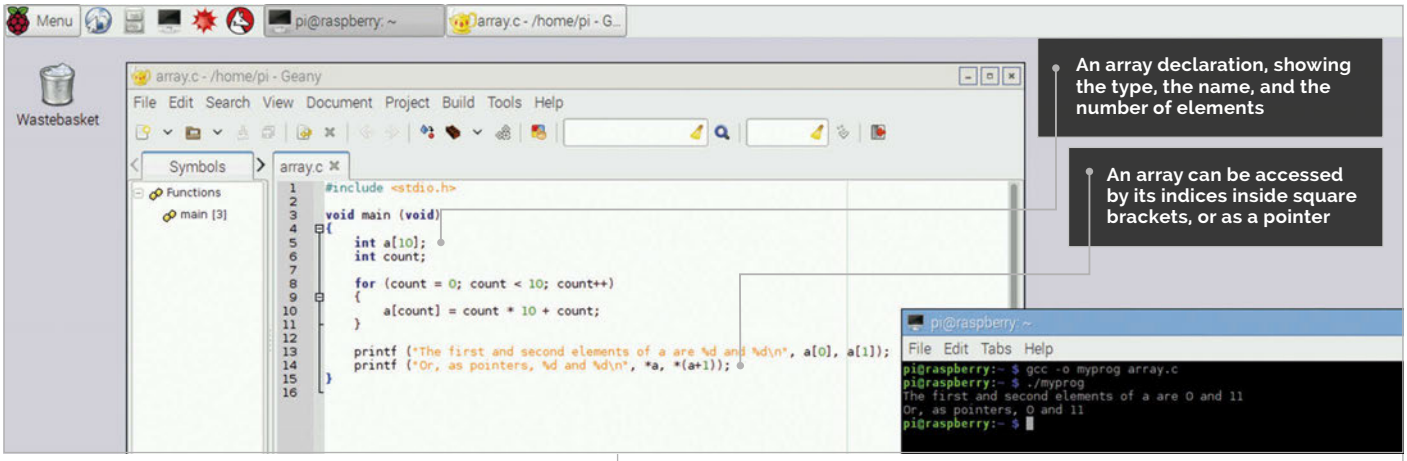
```
int a[10] = { 0, 11, 22, 33, 44, 55, 66, 77, 88, 99 };
```

This only works when the array is first declared; once it exists, you can't use this shortcut and will need to iterate through the array, setting each value in turn.

## Strings

In C, a string is an array of single characters. A character is a specific type in C, called **char** – this holds a single byte, storing an ASCII character. A string with ten bytes would be:

```
char mystring[10];
```


An array declaration, showing the type, the name, and the number of elements
An array can be accessed by its indices inside square brackets, or as a pointer
Wastebasket
array.c - /home/pi - Geany
File Edit Search View Document Project Build Tools Help
Symbols
array.c
Functions
main [3]
#include <stdio.h>
void main (void)
{
int a[10];
int count;
for (count = 0; count < 10; count++)
{
a[count] = count * 10 + count;
}
printf ("The first and second elements of a are %d and %d\n", a[0], a[1]);
printf ("Or, as pointers, %d and %d\n", *a, *(a+1));
}
pi@raspberry: ~
File Edit Tabs Help
pi@raspberry:~ $ gcc -o myprog array.c
pi@raspberry:~ $ ./myprog
The first and second elements of a are 0 and 11
Or, as pointers, 0 and 11
pi@raspberry:~ $

Or, to initialise it at the same time:

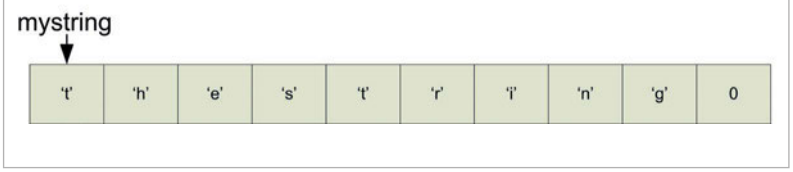
```
char mystring[10] = "thestring";
```

A string in C must end with a byte set to zero, and the memory required to hold this final zero (called the string terminator) must be allocated when you declare the string. So **mystring**, declared as an array of ten chars, can only hold nine or fewer letters..

You can use the index in square brackets to access individual characters in a string, or you can use a pointer. Here's an example using pointers to join two strings together:

```
#include <stdio.h>
void main (void)
{
      char str1[10] = "first";
      char str2[10] = "second";
      char str3[20];
      char *src, *dst;
      src = str1;
      dst = str3;
      while (*src != 0)
      {
         *dst = *src;
         src++;
         dst++;
      }
      src = str2;
      while (*src != 0)
      {
         *dst = *src;
         src++;
         dst++;
      }
      *dst = 0;
      printf ("%s + %s = %s\n", str1, str2, str3);
}
```

We initialise two strings, **str1** and **str2**, and we allocate an empty string, **str3**. We create a pair of **char** pointers, and point **src** at the start of **str1**




**Above** Strings are stored as an array of single characters, with the element after the last character set to zero

and **dst** at the start of the empty **str3**. We loop, copying from **src** to **dst** and incrementing both pointers, until we find the zero that terminates **str1**. We then point **src** at **str2**, and do the same thing again until we find the zero at the end of **str2**. Finally, we write a zero to the end of **str3** to terminate it.

Note the format specifier used to print strings – **%s** is used to print a string, and will display every character from the pointer supplied as an argument up to the first terminating zero. (To print a single character, you can use the format specifier **%c**.)

## Writing to strings

Because the name of a string variable is only a pointer to the first character of the string, you can't use an equals sign to set the value of a string. You can initialise a string variable at the time you declare it, as above, but to set it later, the easiest way is the **sprintf** function. This is like **printf**; the difference is that the first argument it takes is the name of a string variable, and it writes to that instead of to the command line:

```
#include <stdio.h>
void main (void)
{
    int val = 12;
    char string[50];

    sprintf (string, "The value of val is %d\n", val);
    printf ("%s", string);
}
```

The **sprintf** function adds the terminating zero at the end of any string it creates.

### KEEP INSIDE YOUR STRING

As with arrays, it's even easier to write off the end of a string and to corrupt memory elsewhere as a result. If in doubt, declare your string variables with more space than you think you'll need, and make sure that what you are writing into them actually fits.

### NAMES ARE POINTERS

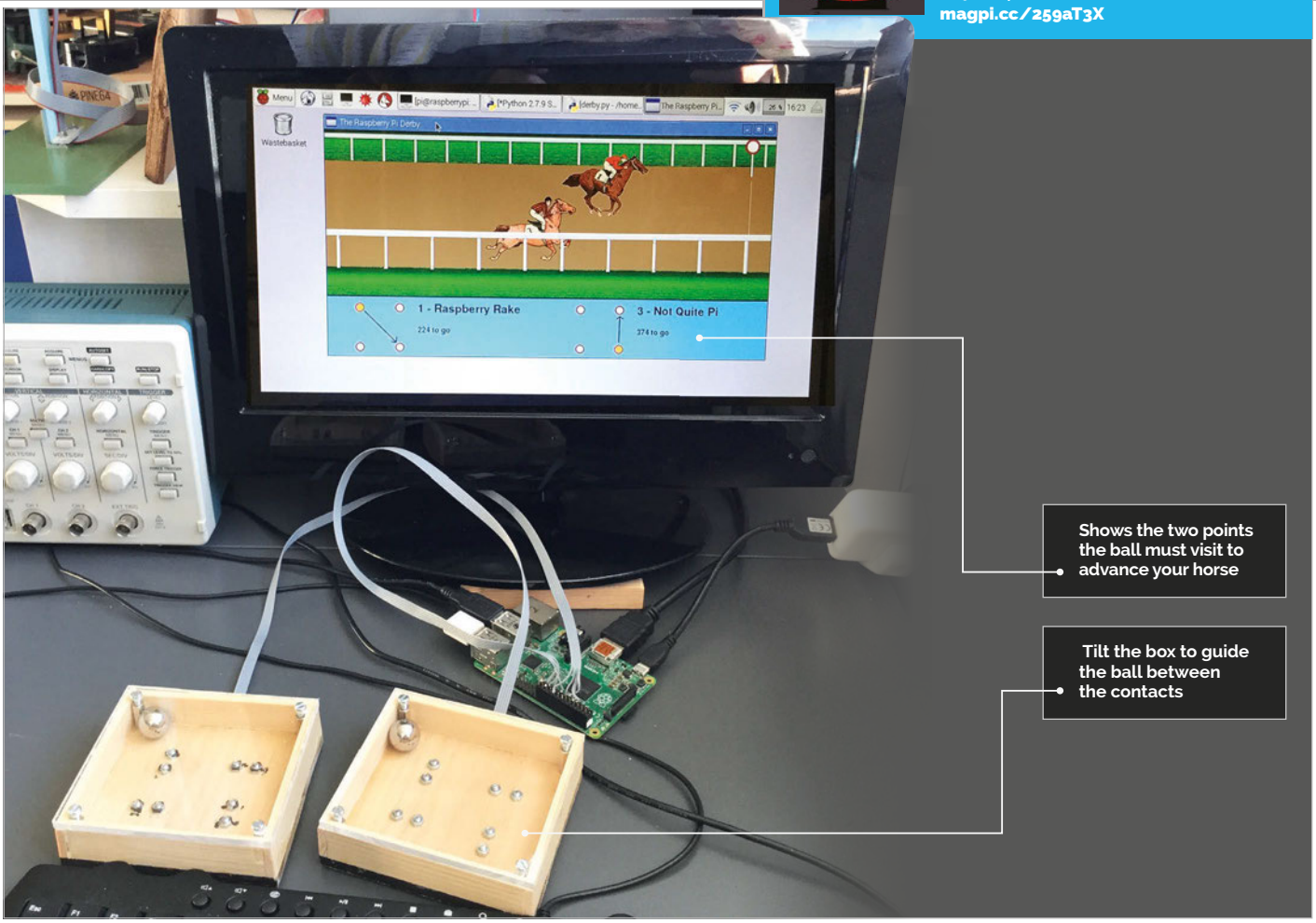
Remember that the name of an array or a string is just a pointer to the first element of the array or string in question, and can be used in the same way as any other pointer; it can be incremented and decremented, or dereferenced to find the value to which it points.

# MIKE'S PI BAKERY

**MIKE COOK**

Veteran magazine author from the old days and writer of the Body Build series. Co-author of *Raspberry Pi for Dummies*, *Raspberry Pi Projects*, and *Raspberry Pi Projects for Dummies*.

**magpi.cc/259aT3X**


1 - Raspberry Rake
3 - Not Quite Pi
Shows the two points the ball must visit to advance your horse
Tilt the box to guide the ball between the contacts

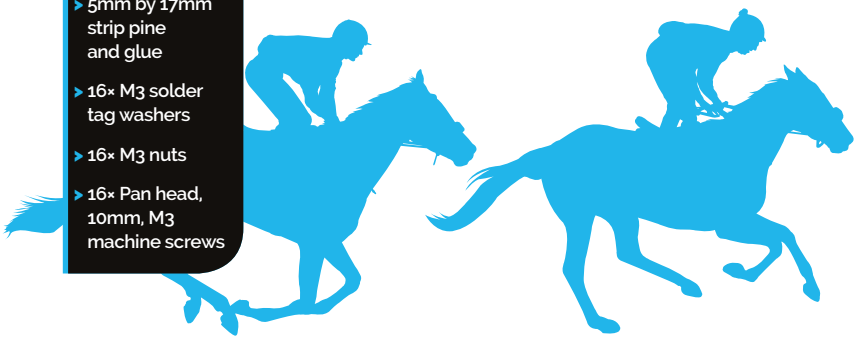
## You'll Need

- 2× 16mm ball bearings
- 8× 15mm M3 hex pillars
- Five-way ribbon cable
- Perspex and 6mm plywood
- 5mm by 17mm strip pine and glue
- 16× M3 solder tag washers
- 16× M3 nuts
- 16× Pan head, 10mm, M3 machine screws

# THE PI DERBY

## A TWO-PLAYER RACING GAME OF SKILL AND DEXTERITY

Will 'Raspberry Rake' or 'Not Quite Pi' be the first to the winning line?



We remember the 'roll a ball' horse race game from amusement parks with great fondness, so this month we thought we would create a two-player race game inspired by it. In the original you had to roll balls up a ramp into a hole and, depending on what hole the ball dropped into, your horse advanced a certain distance. You could have up to 20 wooden horses racing each other, with a prize for the winner. When the game was reset, all the horses moved backwards to the starting line again.

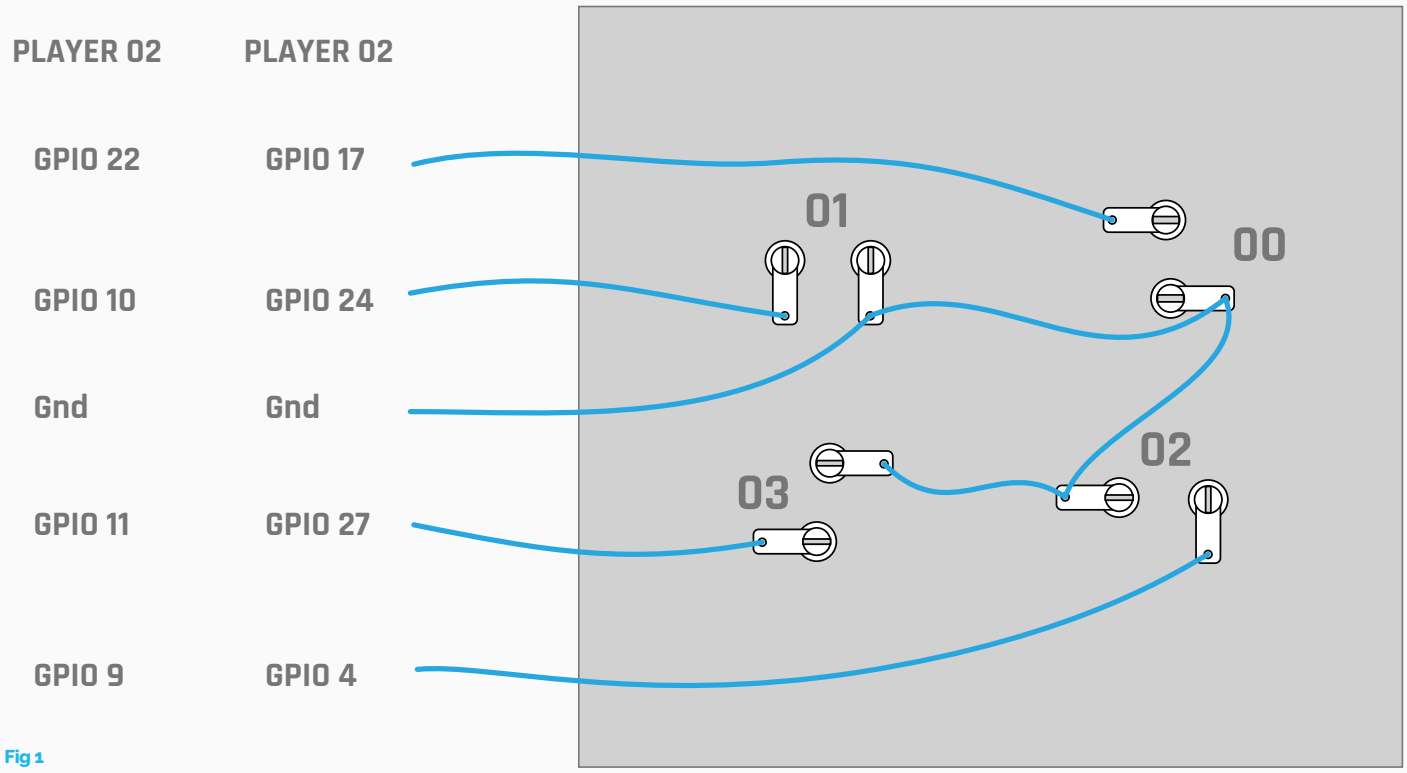



Fig 1

## Our game

As the original attraction was very big, things had to be scaled down drastically to fit in the Bakery. First off, the number of players has been reduced to just two, and the wooden horses have been transferred to on-screen graphics. Finally, in order to reduce the size even further, the skill element has been miniaturised as a small ball bearing puzzle. Each time the ball bearing is manoeuvred between two requested positions, your horse advances, and the first to the finishing line wins. Unlike other race games we've made in the Pi Bakery, this one isn't against the clock but another opponent. The increasing excitement and pressure of the race makes precise control of the ball bearing increasingly difficult, and that's not allowing for unsportsmanlike nudges from the other player.

## The interface

The game interface is simple enough. There are four pairs of nuts in the playing box and as the ball bearing is brought to rest against a pair of them, it completes a circuit and grounds a pulled-up GPIO pin. This is then read by the Raspberry Pi and if it's the position you've been asked to reach, your horse advances. The wiring diagram is shown in **Fig 1**; as this is a two-player game, you will need to make two of them. We avoided the pins GPIO 2 and 3 as there's a strong pull up wired to these, so they needed better contact with the ball to register a touch than the other pins. The alignment of the contact nuts was made so that it was more difficult to go from one position to the other, thus requiring more skill. The construction details are shown in the step-by-step guide.

## BUILDING THE GAME BOXES

### >STEP-01

**Gathering the materials**

The two boxes are 100mm by 90mm, and you need to cut this from 6mm plywood. Also, cut an identical size of acrylic sheet for the cover. Clamp the two together when drilling the pillar holes to make sure they line up. Then cut eight lengths of 5mm by 17mm strip pine, 90mm long, for the sides.

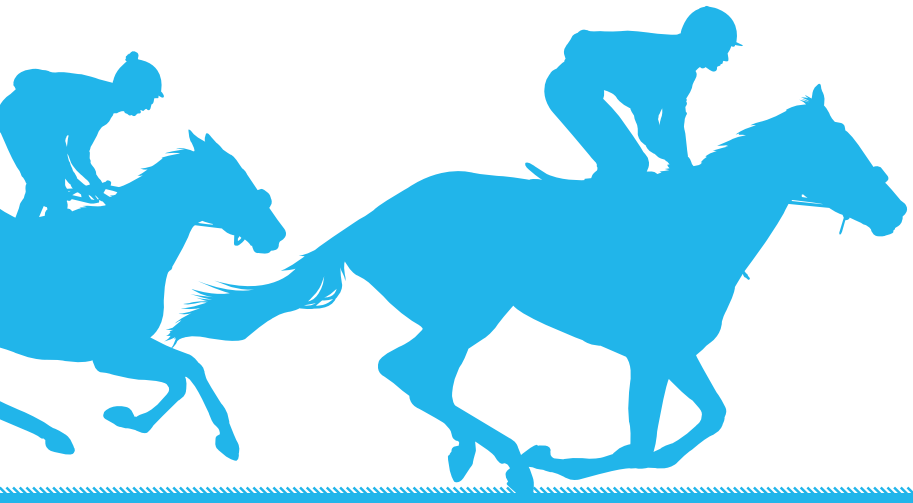
**>STEP-02**

**Make the box bottoms**

A band clamp was used to ensure the sides were perpendicular to the base as they were glued together. We used Gorilla glue and moistened the wood before applying the glue. After about two hours, the clamp was removed and excess glue cut off with a scalpel. Drill the base with an M3 drill for both the corner hex pillars and the nuts and bolts for the ball rests.

## The resources

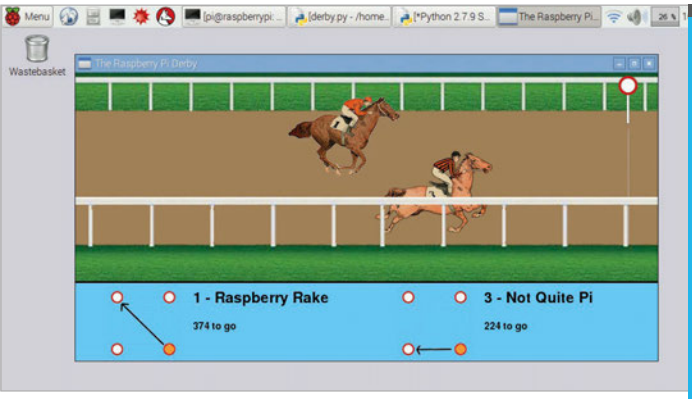
The graphics for the horses were taken from a cut-out supplement, printed in the *Boston Sunday Globe* in September 1905, by RF Ayers; this is now well out of copyright. This cut-out featured two horses in mid-gallop, curiously carrying the numbers 'one' and 'three' on the saddle cloth. We have named the first horse 'Raspberry Rake' and the third 'Not Quite Pi', which we thought were suitable racing names.



These horses were cut out electronically in a graphics package, and the background made transparent. Then they were reduced to just 196 by 136 pixels. The graphic of a track was drawn with a finishing post and line, and then a front rail was drawn with transparent sections between the rails. Finally, a sequence of graphics was drawn to indicate the required path of the ball bearing for each challenge. We also gathered a series of sound effects from the internet to help the game along.

## Playing the game

The graphic by the horse name shows the two points the ball bearing must visit to advance the horse. The filled orange circle indicates the current target for your ball: only when you visit the start position, and then the end one, will the horse move. When you have just visited the start position, there will be a horse whinnying sound, and then you get the horse galloping sound and animation when you visit the last position. When the horse advances, not only does it go forward, but it also moves slightly up and down in a galloping motion; this mimics the original fairground version. When one of the horses crosses




**Above** Move the ball bearing as shown to advance your horse

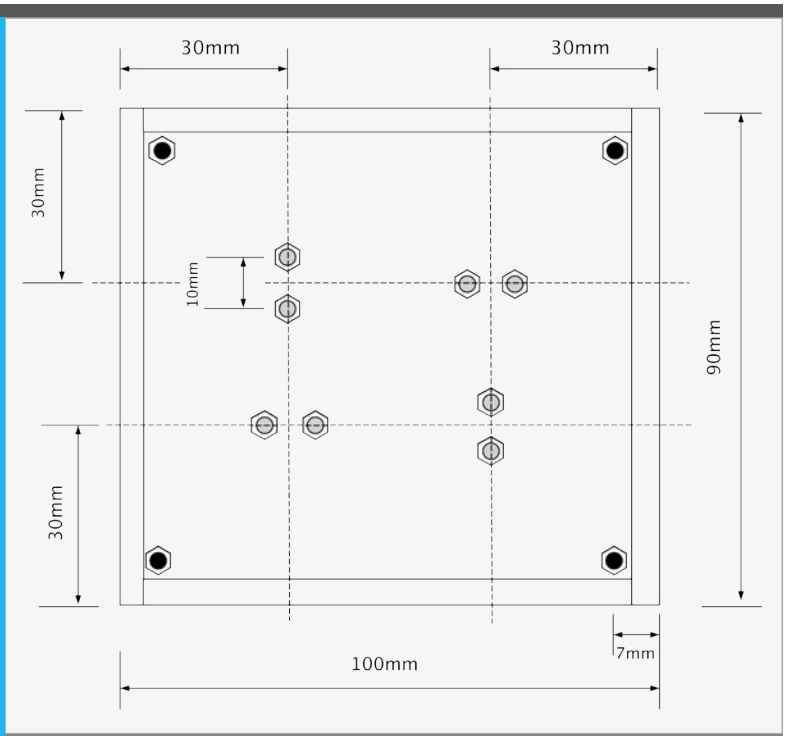
the finishing line, the winning margin is calculated and printed using the correct horse racing terms: a head, short head, length, and so on. Then the horses are wound back to the start with an accompanying ratchet sound just like the original, and the race can start again. At any time, pressing the **RETURN** key will restart the race. We found occasionally that a ball would sit on the contacts but still not make an electrical connection. In that case, a small vibration of the box soon made a good connection.

## The software

The **derby.py** code listing is shown overleaf; this and the accompanying graphics are on our GitHub page (**magpi.cc/1NqJjmV**). Like a lot of our projects, it's written with the Pygame framework, and follows a pattern familiar to regular readers. The **`showPicture`** function first draws the track, then the two horses, and then the front rails; the last bit is the horse's name and the distance to the finishing line, along with the current state of where the ball should go. You'll notice that the variables describing some positions have two or three simple additions, rather than a single number. This makes it a lot easier to spot where the individual offsets are applied when developing the code, but you could pre-add them if it worries you. The **`movePhase`** variable list shows if the current target for the ball is at the start or the end of the line. The horse is checked at each movement to see if it has crossed the winning line, and there's also provision for the unlikely event of a dead heat.

## Taking it further

You can modify the amount of movement a horse makes each time, thus requiring more or fewer ball movements per game. The game could easily be extended to more players; with each player needing four GPIO ports, there are ample resources for six players without resorting to any extra electronics. You could replace the single horse graphic with a number of frames to make smoother, more realistic movement like we did in the Olympic Swimming game (*The MagPi* #48); however, the original idea was to reproduce the mechanical fairground attraction.




## >STEP-03

### Drill the base

The diagram shows the view into the tray; you might be better off marking the back and drilling from that side. Wire up the nuts and bolts to the Raspberry Pi's GPIO lines, using the solder tag washers, and test the connections are correct before fitting the acrylic cover. We used a small piece of double-sided sticky foam to fix the ribbon cable to the back of the box.

## >STEP-04

### Finishing off

Make sure to clean off the ball bearings with some solvent, as finger grease can result in the ball failing to make a good contact. You can paint around the nuts with Bare Conductive paint to make a more reliable contact if you wish, but allow at least a day for this to dry.

# derby.py

```
# Pi Derby - Horse race game
# By Mike Cook - November 2016

import pygame, time, os, random
import wiringpi2 as io

pygame.init()  # initialise graphics interface
pygame.mixer.quit()
pygame.mixer.init(frequency=22050, size=-16,
channels=2, buffer=512)

os.environ['SDL_VIDEO_WINDOW_POS'] = 'center'
pygame.display.set_caption("The Raspberry Pi Derby")
pygame.event.set_allowed(None)
pygame.event.set_allowed([pygame.KEYDOWN,pygame.QUIT])
screen = pygame.display.set_mode([942,466],0,32)
textHeight = 36 ; textHeight2 = 24
font = pygame.font.Font(None, textHeight)
font2 = pygame.font.Font(None, textHeight2)

random.seed() ; winningPost = 704
ballPins = [ [17,24,4,27],[22,10,9,11 ] ]
targetGx = [ 18,102,18,102 ]
targetGy = [ 18,18,102,102 ]
gallopInc = [1,1] ; gallop= [0,0]
dInc = [0,0] # distance increment
puzzle = [5,5] ; restart = True
moveTarget = [1,1] ; movePhase = [0,0]

def main():
  global restart,dInc,gallop,puzzle,moveTarget,movePhase
  initGraphics()
  initGPIO()
  #print "The Pi Derby"
  distance = [-120, -120]
  showPicture(distance,gallop)
  while True:
    checkForEvent()
    if restart :
      gameSound[3].play()
      windBack(distance)
      pygame.mixer.fadeout(1000)
      time.sleep(3.5)
      distance[0] = -120 ; distance[1] = -120
      gallop[0] = 0 ; gallop[1] = 0
      gallopInc[0] = 1 ; gallopInc[1] = 1
      dInc[0] = 0 ; dInc[1] = 0
      puzzle = [5,5]
      moveTarget = [1,1] ; movePhase = [0,0]
      gameSound[0].play()
      showPicture(distance,gallop)
      time.sleep(5)
      restart = False # show Puzzle
      showPicture(distance,gallop)

    moveDetect(distance)
    if dInc[0] or dInc[1] :
      for n in range(0,2):
        if dInc[n] :
          distance[n] += 3
          if dInc[n] :
            dInc[n] -=1
            gallop[n] = gallopAdv(gallop,n)
      showPicture(distance,gallop)
      if checkForFinish(distance):
        restart = True
        time.sleep(4.0)

def moveDetect(dis):
  global dInc, puzzle, moveTarget
  move = -1
  for n in range(0,2):
    move = checkInput(n) # look at ball switch
    if move != -1 :
    # check if it is a valid move
    if move ==  moveTarget[n] :
      if movePhase[n] == 1 : # move complete
        dInc[n] = 50 # move this * 3
        gameSound[1].play()
        last = moveTarget[n]
        puzzle[n] = random.randint(0,11)
        while last == moveState[puzzle[n]][0] :
          puzzle[n] = random.randint(0,11)
        moveTarget[n] = moveState[puzzle[n]][0]
        movePhase[n] = 0
      else :
        moveTarget[n] = moveState[puzzle[n]][1]
        movePhase[n] = 1
        showPicture(dis,gallop) # update move graphic
        if n ==0 :
          gameSound[4].play()
        else :
          gameSound[5].play()

def gallopAdv(gallop,n):
  global gallopInc
  gallop[n] += gallopInc[n]
  if gallop[n] > 6 or gallop[n] <0:
    gallopInc[n] = -gallopInc[n]
  return gallop[n]

def checkForFinish(d):
  if d[0] >=winningPost or d[1] >=winningPost:
    gameSound[2].play()
    finish(d[0] - d[1])
    time.sleep(3.0)
    return True
  else :
    return False

def windBack(d):
  wind = 4
  while d[0] >= -120 or d[1] >= -120:
    showPicture(d,gallop)
    checkForEvent()
    for n in range(0,2):
      if d[n] >=-120:
        d[n] -= wind

def checkInput(player):
  ball = -1
  for pin in range(0,4):
    if io.digitalRead(ballPins[player][pin]) == 0 :
      ball = pin
  return ball
```

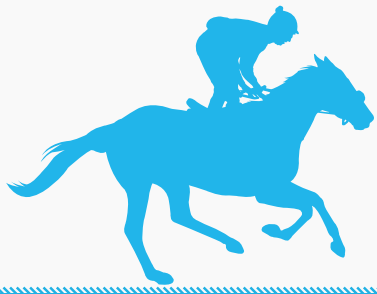
Language
>PYTHON 2

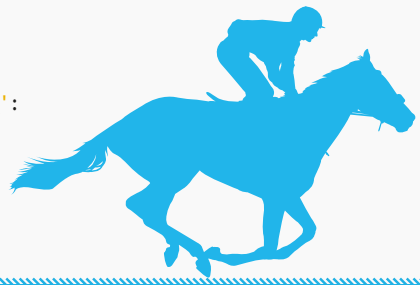
DOWNLOAD:
magpi.cc/1NqJjmV

PROJECT VIDEOS

Check out Mike's Bakery videos at:
magpi.cc/1NqJnTz

```
125.
126. def initGPIO():
127.   try :
128.     io.wiringPiSetupGpio()
129.   except :
130.     print"start IDLE with 'gksudo idle' from command line"
131.     os._exit(1)
132.   for player in range(0,2):
133.     for pin in range (0,4):
134.       io.pinMode(ballPins[player][pin],0)
135.       # input enable pull up
136.       io.pullUpDnControl(ballPins[player][pin],2)
137.
138. def showPicture(run,gal):
139.   screen.blit(background,[0,0])
140.   screen.blit(horse1,[run[0],30+gal[0] ] )
141.   screen.blit(horse3,[run[1],120+gal[1] ] )
142.   screen.blit(foreground,[0,200])
143.   pygame.draw.rect(screen,(102,204,255), (0,340,942,136), 0)
144.   if not restart :
145.     screen.blit(move[puzzle[0]],[50,345])
146.     pygame.draw.circle(screen, (255,154,51),(
    targetGx[moveTarget[0]]+50,targetGy[moveTarget[0]]+345),8,0)
147.     screen.blit(move[puzzle[1]],[471+50,345])
148.     pygame.draw.circle(screen, (255,154,51),(
    targetGx[moveTarget[1]]+50+471,targetGy[moveTarget[1]]+345),8,0)
149.   drawWords("1 - Raspberry Rake",120+70,350,0)
150.   drawWords("3 - Not Quite Pi",120+471+70,350,0)
151.   drawWords(str(winningPost-run[0])+" to go",120+70,400,1)
152.   drawWords(str(winningPost-run[1])+" to go",120+70+471,400,1)
153.   pygame.display.update()
154.
155. def finish(distance):
156.   if distance != 0:
157.     caption = "Wins by " + margin(abs(distance))
158.     if distance < 0 :
159.       x= 120+70+471
160.     else:
161.       x= 120+70
162.     drawWords(caption,x,400,1)
163.   else :
164.     drawWords("Dead heat",120+70,400,1)
165.     drawWords("Dead heat",120+70+471,400,1)
166.   pygame.display.update()
167.
168.
169. def margin(dist):
170.   separation = dist / 120.0
171.   if separation >1.8:
172.     return "a distance"
173.   elif separation >0.9:
174.     return "a length"
175.   elif separation >0.7:
176.     return "three quarters of a length"
177.   elif separation >0.4:
178.     return "half a length"
179.   elif separation >0.25:
180.     return "a neck"
181.   elif separation >0.2:
182.     return "a short neck"
183.   elif separation >0.1:
184.     return "a head"
185.   elif separation >0.05:
186.     return "a short head"
187.   else :
188.     return "a nose"
189.
190. def drawWords(words,x,y,f) :
191.   if f == 0:
192.     th = textHeight
193.   else :
194.     th = textHeight2
195.   textSurface = pygame.Surface(
    (14,th))
196.   textRect = textSurface.get_rect()
197.   textRect.left = x
198.   textRect.top = y
      pygame.draw.rect(screen,(
199. 102,204,255),(x,y,14,th-10), 0)
200.   if f==0 :
        textSurface = font.render(words, True, (0,0,0), (102,204,255))
201.   else :
202.     textSurface = font2.render(
    words, True, (0,0,0), (102,204,255))
203.   screen.blit(textSurface,textRect)
204.
205. def initGraphics():
206.   global background,foreground,horse1,horse3,move,moveState,
    gameSound
207.   soundEffects = ["start","gallop","end","ratchet","horse1",
    "horse2"]
208.   background =  pygame.image.load(
    "images/track.png").convert_alpha()
209.   foreground =  pygame.image.load(
    "images/rail.png").convert_alpha()
210.   horse1 =  pygame.image.load("images/rr.png").convert_alpha()
211.   horse3 =  pygame.image.load("images/np.png").convert_alpha()
212.   move = [ pygame.image.load(
    "images/dir"+str(m)+".png").convert_alpha() for m in range(0,12) ]
213.   moveState = [ [0,3],[3,0],[2,1],[1,2],[0,1],[1,0],[0,2],[2,3],
    [3,1],[1,3],[3,2],[2,0] ]
214.   gameSound = [ pygame.mixer.Sound(
    "sounds/"+soundEffects[sound]+".wav") for sound in range(0,6)]
215.
216. def terminate(): # close down the program
217.    print "Closing down please wait"
218.    pygame.mixer.quit()
219.    pygame.quit() # close pygame
220.    os._exit(1)
221.
222. def checkForEvent(): # see if we need to quit
223.    global restart
224.    event = pygame.event.poll()
225.    if event.type == pygame.QUIT :
226.      terminate()
227.    if event.type == pygame.KEYDOWN :
228.      if event.key == pygame.K_ESCAPE :
229.        terminate()
230.      if event.key == pygame.K_RETURN :
231.        restart = True
232.
233. # Main program logic:
234. if __name__ == '__main__':
235.    main()
```

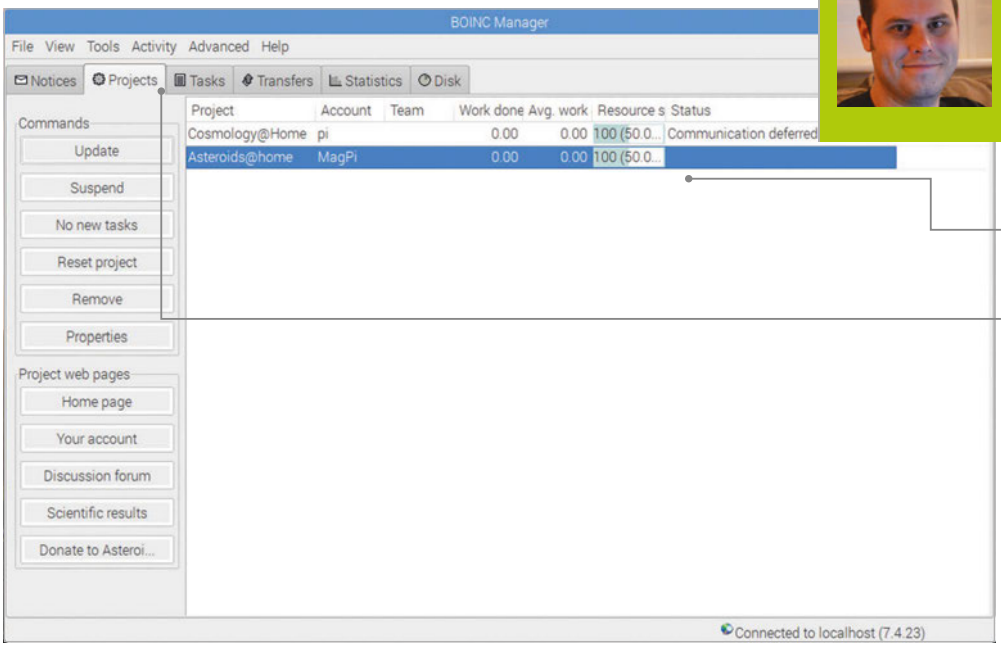




**ROB ZWETSLOOT**

Tinkerer, sometime maker, other-times cosplayer, and all-the-time features editor of *The MagPi*.

**magpi.cc / @TheMagPi**




# SHARE YOUR POWER WITH BOINC

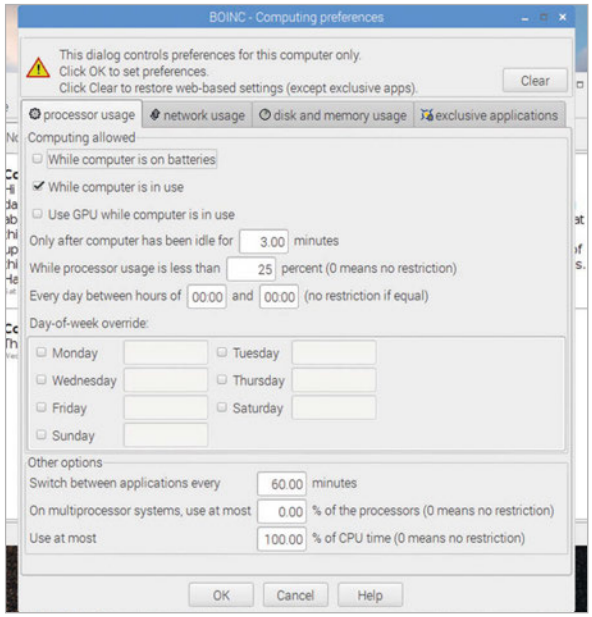
## You'll Need

- BOINC **boinc.berkeley.edu**
- A Raspberry Pi that's on a lot
- An internet connection

Got a Raspberry Pi that idles away most of its day? Lend some of its CPU power to science

Remember screen savers? It occurred to us the other day that they've gone almost totally out of fashion, with turning your screen off being the preferred (and probably greener) solution to burn-in. One of the more popular screen savers of the time was a little display that showed number crunching from SETI, the organisation that looks for extraterrestrial life. With this screen saver, you'd also be lending some of your computer power to help them out.

The technology still exists with BOINC and is used in many more applications these days, such as protein folding and anything else that requires lots of processing power. We recently had a reader write in about using the idling CPU power of a Pi with it, so we decided to put together a little tutorial to let people share some Pi goodness with the science world.


BOINC - Computing preferences
This dialog controls preferences for this computer only.
Click OK to set preferences.
Click Clear to restore web-based settings (except exclusive apps).
Clear
processor usage
network usage
disk and memory usage
exclusive applications
Computing allowed
While computer is on batteries
While computer is in use
Use GPU while computer is in use
Only after computer has been idle for 3.00 minutes
While processor usage is less than 25 percent (0 means no restriction)
Every day between hours of 00:00 and 00:00 (no restriction if equal)
Day-of-week override:
Monday
Tuesday
Wednesday
Thursday
Friday
Saturday
Sunday
Other options
Switch between applications every 60.00 minutes
On multiprocessor systems, use at most 0.00 % of the processors (0 means no restriction)
Use at most 100.00 % of CPU time (0 means no restriction)
OK
Cancel
Help

**Right** Change the settings to make sure your Pi works as best as it can with BOINC

## >STEP-01

### Choosing a Pi

Not just any Raspberry Pi is suitable for running BOINC. If you're making projects and using your Pi as a normal computer, you're likely not going to have it idling often. The kind of Pi that works best for BOINC is something that's on all the time, or at least for a long time each day: file servers, home automation, Christmas tree lights, and so on. These have a lot of processing power to spare, more than enough to lend some to another project using BOINC.

Remember, though: this does mean it will use up a little more electricity while performing BOINC tasks, so keep that in mind.

## >STEP-02
### Installing BOINC

BOINC is included as one of the packages available for Raspbian, so installing it is pretty easy. First, you need to perform the customary update to the Raspberry Pi by opening a Terminal and entering:

```
sudo apt-get update
sudo apt-get upgrade
```

Once that's complete, you can then install BOINC from the command line. Do this with:

```
sudo apt-get install boinc
```

This will install BOINC and the graphical BOINC Manager tool for easily accessing and controlling how you use BOINC. It even shows stats for your projects.

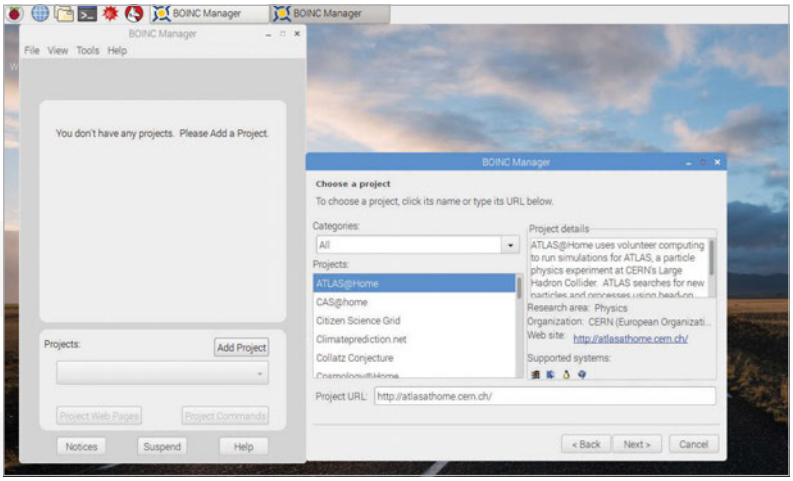
## >STEP-03
### Open BOINC

BOINC works in both the command line and on the graphical desktop. It's a bit more involved to get it working in the command line, but if that's your preferred method, we suggest taking a look at this great tutorial on how to do so: **magpi.cc/2hgkTYO**.

For this tutorial, we're going to be working in the desktop environment. Go to the top-left menu in Raspbian; under System Tools you'll find the BOINC Manager. Open it up, and we can start choosing and adding projects to the system.

## >STEP-04
### Choose a project

Choose 'Add project' on the screen and then go to Next. If you already have a BAM! or GridRepublic account, you can also add it here instead. You'll then be presented with a list of projects you can join, such as the aforementioned SETI. They'll usually give details on what they're studying and how you can help them, but you might need to do some extra online research as well.




**Above** Choose one or more of multiple projects that will help scientists

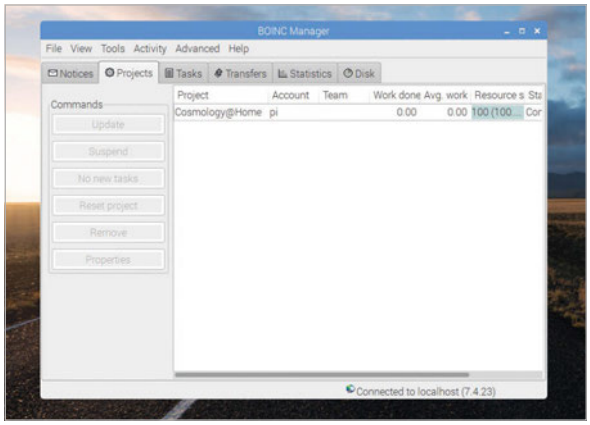
Click your choice and then press Next. From here you need to create an account (or log in if you already have one), and the project will be ready to start.

## >STEP-05
### Configure BOINC

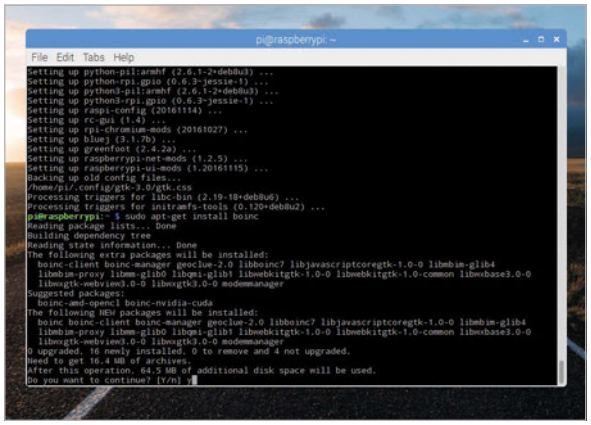
The default settings for BOINC are fine, but you might want to do a bit of tweaking anyway, especially to change what it considers as idle on your system. From the View menu, select Advanced View to access all the options. Here you can change not only the processor usage, but also set limits on network and disk/memory usage. You can even drill down a bit further into what tasks BOINC is running, how many resources it's using, and further stats.

## >STEP-06
### Run BOINC

All you need to do now is leave your Raspberry Pi and BOINC running. It will automatically take tasks from the project you signed up to and do some crunching on them. It may take a little while longer on the Pi than some other computers, but you'll be doing your bit for science nonetheless. After reboots, you may need to double-check it's running, and it's best to check Pis that are on 24/7 every now and then to make sure they're doing the job properly as well.




**Above** The advanced view gives you more options for keeping an eye on your projects


pi@raspberrypi: ~
File Edit Tabs Help
Setting up python-pil:armhf (2.6.1-2+deb8u3) ...
Setting up python-rpi.gpio (0.6.3~jessie-1) ...
Setting up python3-pil:armhf (2.6.1-2+deb8u3) ...
Setting up python3-rpi.gpio (0.6.3~jessie-1) ...
Setting up raspi-config (20161114) ...
Setting up rc-gui (1.4) ...
Setting up rpi-chromium-mods (20161027) ...
Setting up bluej (3.1.7b) ...
Setting up greenfoot (2.4.2a) ...
Setting up raspberrypi-net-mods (1.2.5) ...
Setting up raspberrypi-ui-mods (1.20161115) ...
Backing up old config files...
/home/pi/.config/gtk-3.0/gtk.css
Processing triggers for libc-bin (2.19-18+deb8u6) ...
Processing triggers for initramfs-tools (0.120+deb8u2) ...
pi@raspberrypi:~ $ sudo apt-get install boinc
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
boinc-client boinc-manager geoclue-2.0 libboinc7 libjavascriptcoregtk-1.0-0 libmbim-glib4
libmbim-proxy libmm-glib0 libqmi-glib1 libwebkitgtk-1.0-0 libwebkitgtk-1.0-common libwxbase3.0-0
libwxgtk-webview3.0-0 libwxgtk3.0-0 modemmanager
Suggested packages:
boinc-amd-opencl boinc-nvidia-cuda
The following NEW packages will be installed:
boinc boinc-client boinc-manager geoclue-2.0 libboinc7 libjavascriptcoregtk-1.0-0 libmbim-glib4
libmbim-proxy libmm-glib0 libqmi-glib1 libwebkitgtk-1.0-0 libwebkitgtk-1.0-common libwxbase3.0-0
libwxgtk-webview3.0-0 libwxgtk3.0-0 modemmanager
0 upgraded, 16 newly installed, 0 to remove and 4 not upgraded.
Need to get 16.4 MB of archives.
After this operation, 64.5 MB of additional disk space will be used.
Do you want to continue? [Y/n] y

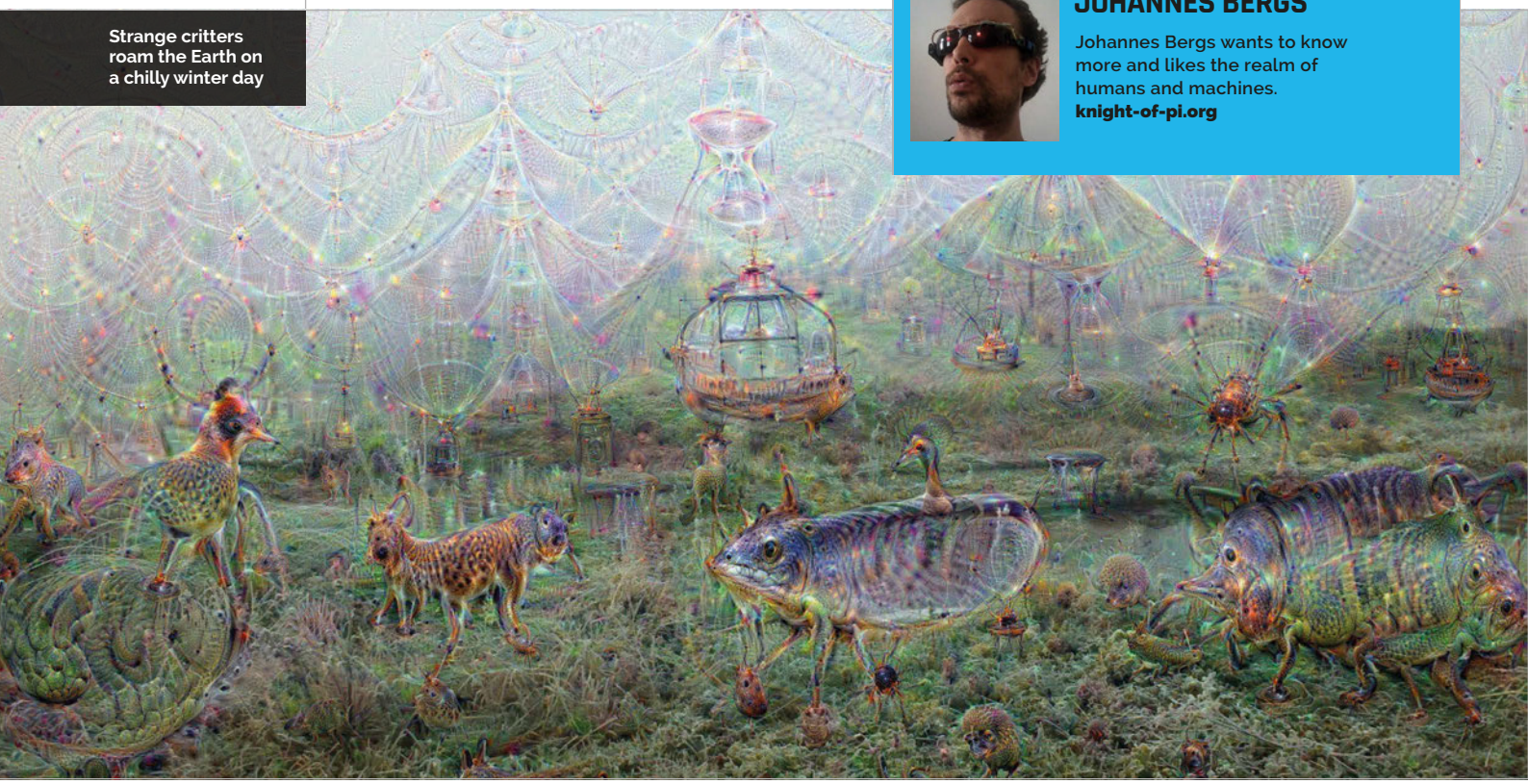
**Above** The installation on the command line is quick and pretty easy – you'll have it in no time

### ENERGY USAGE

As mentioned, you'll use up more electricity using BOINC. Change the settings to tweak how much it uses.

### SENSITIVITY

Worried your Pi will want to do something and not have the power? Tweak how it defines idle to make sure the Pi gets power when it needs it.



Strange critters roam the Earth on a chilly winter day

**JOHANNES BERGS**

Johannes Bergs wants to know more and likes the realm of humans and machines.
**knight-of-pi.org**


# GOOGLE DEEPDREAM ON THE RASPBERRY PI

Do computers dream of electric sheep? Find out by making your Raspberry Pi dream with the amazing Google DeepDream algorithm

## You'll Need

- Camera Module **magpi.cc/28ljlsz**
- PsyCam **magpi.cc/2eCSQOD**

### RASPBERRY PI 2 COMPATIBILITY

Although the Pi 3 with its advanced processing power is preferable, the project will run on a Pi 2 without modifications.

Dreams can be beautiful, surreal, or even scary. While we can assume that they play a role in learning, their exact purpose remains unknown. What would a computer see if it could dream? You can find out by installing the amazing Google DeepDream software on your Raspberry Pi!

DeepDream is derived from a neural network for object recognition, which has been modified to express every object it vaguely recognises. The resulting images are impressively surrealistic, bearing more than a passing resemblance to the paintings of artists like Salvador Dalí.

Neural networks are a very versatile technology. They can be applied to a variety of complex problems like facial recognition, text translation, or even playing games like Go.

Caffe, the neural network upon which Google built DeepDream, was originally developed for identifying objects in images. To achieve this, the network is trained on a large set of images to identify the main features of the objects presented.

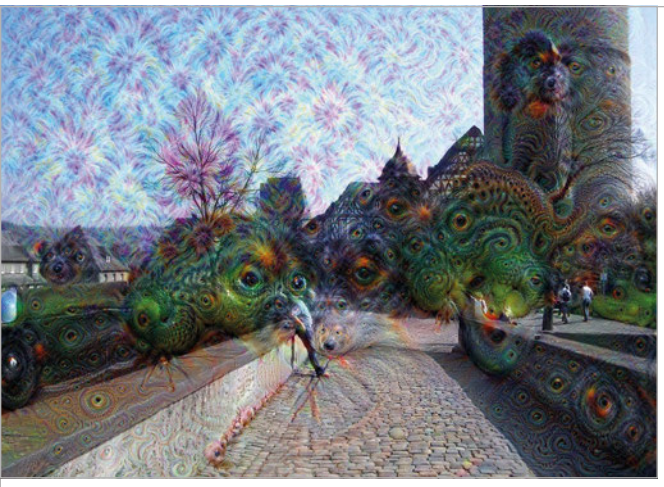
## Artificial neurons

Artificial neurons mimic their natural counterparts, the neurons in the human brain. Every artificial neuron has an activation threshold and numerous weighted incoming and outgoing connections to other neurons. If the sum of the weighted signals from the incoming connections exceeds the activation threshold, the neuron fires a signal.

## Neural networks

Generally, neural networks consist of at least three layers of neurons. The input layer reads the input, one or more hidden layers process the information, and the output layer shows the result.

Consider a neural network for recognising the numbers zero to nine in images with a size of 28×28 pixels. The input layer would have 784 neurons, one for every pixel. If an image is presented to the input layer, every neuron in it produces a signal with a strength corresponding to the greyscale value of its pixel, with a darker pixel generating a stronger signal.



**Above** A magical city in a far-off land

Every input neuron is connected to all neurons of the first hidden layer, but every connection has a distinct weight. The higher the weight, the more of the input neuron's signal reaches the hidden neuron. If the total signal strength arriving at a hidden neuron tops its activation threshold, the hidden neuron fires a signal to all neurons of the next layer which, in this simplified example, is the output layer. Again, the signal intensity depends on the weight of the connection and the incoming signal strength.

The output layer has one neuron for every object to be classified, so ten neurons are used to identify the numbers from zero to nine. If the activation threshold of an output neuron is surpassed by the weighted incoming connections, the resulting signal strength is a measure of confidence in the classification.

## Machine learning

The weights and thresholds are initialised randomly, which can cause very bad classification results.

If an output neuron is activated incorrectly (e.g. if an image from the training set shows a five, but the output neuron for two produces a strong signal), its activation threshold and all the weights of its incoming connections are adjusted. Then, the error is propagated back proportionally through all connected neurons lower in the chain, from the highest hidden layer down to the input layer. This process is repeated for all of the images in the training data set until the results given by the network begin to improve.

Instead of accurately detecting objects in images, by contrast, DeepDream actually changes the input image to make it more similar to the objects learned from the training data.

Imagine a cloud as an input image. Some structure in the cloud might have a vague similarity to the features DeepDream associates with a 'dog-like' object. If this is the case, the input image is changed to look more dog-like.

Additionally, DeepDream allows for selection of the hidden layer depth. Since a layer has more detail the closer it is to the output level, output images can range from basic shapes to detailed dream-like creatures.

## Have some DeepDreams

Insert an SD card with Raspbian Jessie installed into the Raspberry Pi, attach the Camera Module and peripherals, then boot it up. To install the DeepDream software, enter the following in a Terminal:

```
mkdir ~/deepdream && cd ~/deepdream
git clone https://github.com/JoBergs/PsyCam
cd PsyCam
python install_tools.py packages
python install_tools.py caffe
python install_tools.py protobuf
python install_tools.py camera
sudo reboot
```

After downloading the project from GitHub, use the custom installer to first install all packages with **pip** or **apt-get**. Then, install the open-source neural network framework Caffe. Because speed matters, you should also install the serial data processor protobuf from Google. Finally, activate the camera and reboot.

All in all, the installation takes a few hours, so you'll need to be patient! The installer should also work for any newish Ubuntu operating system. If you encounter problems, try using the manual installation instructions at **magpi.cc/2eCSxDt.**

Enter the following, and your Pi can start to dream:

```
cd ~/deepdream/PsyCam
python psycam.py
```

The network parameters depth (**-d**), octave (**-o**), and type (**-t**) are randomized. Add a **-c** to dream continuously. Pass **-i IMAGE.jpg** to use an image as the base for the dream instead of a snapshot. Find more information on input arguments by checking the command-line help:

```
python psycam.py -help
```

When the Pi finishes dreaming, the dream and the original photo are stored in the directory **/home/pi/deepdream/PsyCam/dreams** with a timestamp. You can watch them by opening the directory in the file browser and double-clicking the image.

Alice in a very colourful wonderland

**Language**
>PYTHON

**DOWNLOAD:**
magpi.cc/2eCSQOD

### ORIGINAL DEEPDREAM ARTICLE

Have a look at the amazing original article about DeepDream by Alexander Mordvintsev, Christopher Olah, and Mike Tyka here: **magpi.cc/2eCRHH3**

### NEURAL NETWORK TUTORIAL

If you want to learn more about neural networks, why not check out this interesting and well-written introduction by Michael Nielsen?: **magpi.cc/2eCUO1u**

### KIVY GUI

Try the powerful Python GUI framework Kivy for building a GUI: it is slick enough to embrace DeepDream.

Sonic Pi PART 15

**SAM AARON**

Sam is the creator of Sonic Pi. By day he's a research associate at the University of Cambridge Computer Laboratory; by night he writes code for people to dance to.
**sonic-pi.net**

# SAMPLE STRETCHING

**Sam Aaron** shows how to do more with samples

**You'll Need**

- Raspberry Pi running Raspbian
- Sonic Pi v2.9+
- Speakers or headphones with a 3.5mm jack
- Update Sonic Pi: `sudo apt-get update && sudo apt-get install sonic-pi`

When people discover Sonic Pi, one of the first things they learn is how simple it is to play pre-recorded sounds using the **sample** function. For example, you can play an industrial drum loop, hear the sound of a choir, or even listen to a vinyl scratch, all via a single line of code. However, many people don't realise that you can actually vary the speed at which the sample is played back for some powerful effects and a whole new level of control of your recorded sounds. So, fire up a copy of Sonic Pi and let's get started stretching some samples!

## Slowing samples down

To modify the playback rate of a sample, we need to use the **rate:** opt:

```
sample :guit_em9, rate: 1
```

If we specify a **rate:** of **1**, then the sample is played back at the normal rate. If we want to play it back at half speed, we simply use a **rate:** of **0.5**:

```
sample :guit_em9, rate: 0.5
```

Notice that this has two effects on the audio. Firstly, the sample sounds lower in pitch; secondly, it takes twice as long to play back (see the boxout on the next page for an explanation of why this is the case). We can even choose lower rates moving towards **0**, so a **rate:** of **0.25** is a quarter speed, **0.1** is a tenth of the speed, and so on. Try playing with some low rates and see if you can turn the sound into a low rumble.

## Speeding samples up

In addition to making the sound longer and lower using a small rate, we can use higher rates to make the sound shorter and higher. Let's play with a drum loop this time. First, listen to how it sounds at the default rate of **1**:

```
sample :loop_amen, rate: 1
```

Now, let's speed it up a little:

```
sample :loop_amen, rate: 1.5
```

And we just moved musical genres from old-skool techno to jungle! Notice how the pitch of each drum hit is higher, as well as the whole rhythm speeding up. Now, try even higher rates and see how high and short you can make the drum loop. For example, if you use a rate of **100**, the drum loop turns into a click!

## Reverse gear

I'm sure many of you are thinking the same thing right now: "What if you use a negative number for the rate?". Let's think about this for a moment. If our **rate:** opt signifies the speed at which the sample is played back, with **1** being normal speed, **2** being double speed, and **0.5** being half speed, **-1** must mean backwards! Let's try it on a snare. First, play it back at the normal rate:

```
sample :elec_filt_snare, rate: 1
```

Now, play it backwards:

```
sample :elec_filt_snare, rate: -1
```

Of course, you can play it backwards twice as fast with a rate of **-2**, or backwards at half speed with a rate of **-0.5**. Now, play around with different negative rates and have fun. It's particularly amusing with the **:misc_burp** sample!

## Pitch bending

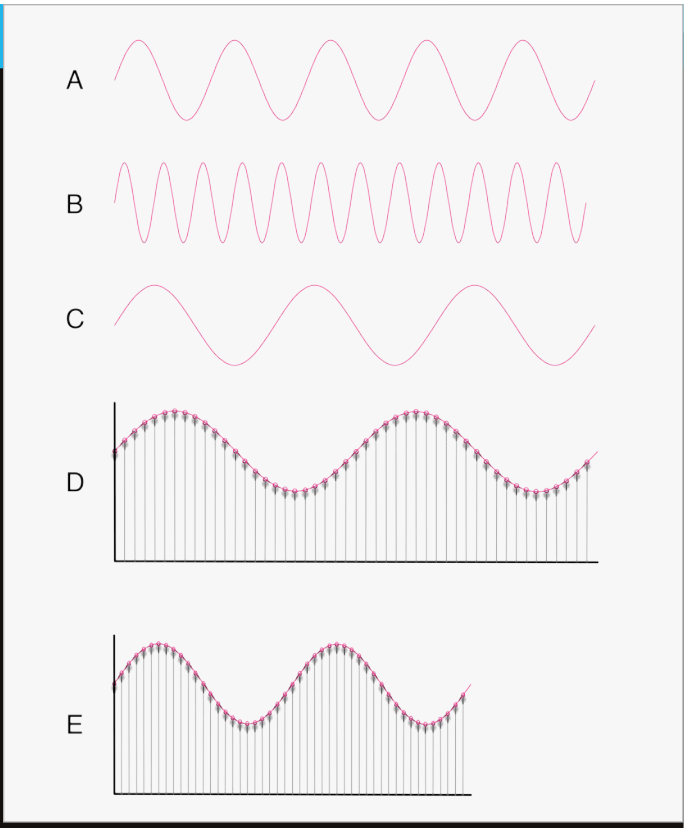
As we've seen, using a faster rate will make the sound higher in pitch, and a slower rate will make the sound lower in pitch. A very simple and useful trick is to know that doubling the rate actually results in the pitch being an octave higher; inversely, halving the rate results in the pitch being an octave lower. This means that for melodic samples, playing it alongside itself at double/half rates actually sounds rather nice:

## SAMPLE, RATE, AND PITCH

One of the effects of rate modification on samples is that faster rates result in the sample sounding higher in pitch, and slower rates result in the sample sounding lower in pitch. Another place you may have heard this effect in everyday life is when you're cycling or driving past a beeping pedestrian crossing. As you're heading towards the sound source, the pitch is higher than when you're moving away from the sound: the so-called Doppler effect. Why is this?

Let's consider a simple beep which is represented by a sine wave. If we use an oscilloscope to plot a beep, we'll see something like **Figure A**. If we plot a beep an octave higher, we'll see **Figure B**, and an octave lower will look like **Figure C**. Notice that the waves of higher notes are more compact, and the waves of lower notes are more spread out.

A sample of a beep is nothing more than a lot of numbers (x and y coordinates) which, when plotted onto a graph, will redraw the original curves. See **Figure D**, where each circle represents a coordinate. To turn the coordinates back into audio, the computer works through each x value and sends the corresponding y value to the speakers. The trick here is that the rate at which the computer works through the x numbers doesn't have to be the same as the rate at which they were recorded. In other words, the space (representing an amount of time) between each circle can be stretched or compressed. So, if the computer walks through the x values faster than the original rate, it will have the effect of squashing the circles closer together, which will result in a higher-sounding beep. It will also make the beep shorter, as we will work through all the circles faster. This is shown in **Figure E**.




Finally, one last thing to know is that a mathematician called Fourier proved that any sound is actually lots and lots of sine waves all combined together. Therefore, when we compress and stretch any recorded sound, we're actually stretching and compressing many sine waves all at the same time in exactly this manner.

```
sample :bass_trance_c, rate: 1
sample :bass_trance_c, rate: 2
sample :bass_trance_c, rate: 0.5
```

However, what if we just want to alter the rate so that the pitch goes up one semitone (one note up on a piano)? Sonic Pi makes this very easy via the **rpitch:** opt:

```
sample :bass_trance_c
sample :bass_trance_c, rpitch: 3
sample :bass_trance_c, rpitch: 7
```

If you take a look at the log on the right, you'll notice that an **rpitch:** of **3** actually corresponds to a rate of **1.892** and a **rpitch:** of **7** corresponds to a rate of **1.4983**. Finally, we can even combine **rate:** and **rpitch:** opts:

```
sample :ambi_choir, rate: 0.25, rpitch: 3
sleep 3
sample :ambi_choir, rate: 0.25, rpitch: 5
sleep 2
sample :ambi_choir, rate: 0.25, rpitch: 6
sleep 1
sample :ambi_choir, rate: 0.25, rpitch: 1
```

## Bringing it all together

Language >RUBY

Let's look at a simple piece combining these ideas. Copy it into an empty Sonic Pi buffer, press Play, listen to it for a while, and then use it as a starting point for your own piece. See how much fun it is to manipulate the playback rate of samples. As an added exercise, try recording your own sounds and play around with the rate to see what crazy sounds you can make.

```
live_loop :beats do
  sample :guit_em9, rate: [0.25, 0.5, -1].choose, amp: 2
  sample :loop_garzul, rate: [0.5, 1].choose
  sleep 8
end

live_loop :melody do
  oct = [-1, 1, 2].choose * 12
  with_fx :reverb, amp: 2 do
    16.times do
      n = (scale 0, :minor_pentatonic).choose
      sample :bass_voxy_hit_c, rpitch: n + 4 + oct
      sleep 0.125
    end
  end
end
```

**ROB ZWETSLOOT**
Tinkerer, sometime maker, other-times cosplayer, and all-the-time features editor of *The MagPi*.
**magpi.cc / @TheMagPi**

# PISPY CAM

Set up a motion-activated spy camera in your room to find out if anyone's coming in when you're not there

## You'll Need

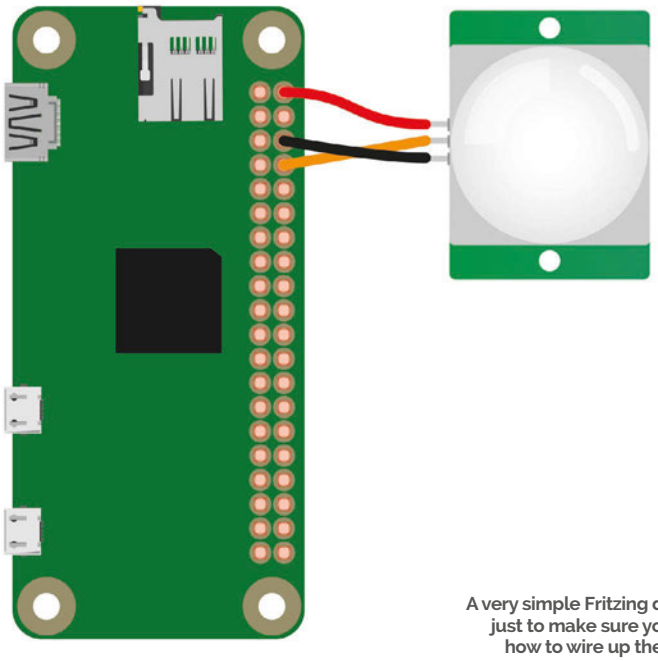
- Pi Zero
- PIR motion sensor **magpi.cc/2gCQKPj**
- Raspberry Pi Camera Module **magpi.cc/28IjIsz**
- Pi Zero camera cable adapter **magpi.cc/2gT2KwE**
- Portable battery

We've all been there. You've gone out for the day and you know you closed your bedroom door, but you come back and it's slightly ajar. Who's been in there? Were they friend or foe?

## >STEP-01

### Connect up the camera

Attaching the camera to the Raspberry Pi Zero is easy, and if you read our camera beginner's guide you'll already have an idea. You'll need to first switch out the cable in the Camera Module with the adapter for the Pi Zero. Gently pull down on the clasp that keeps the ribbon on the Camera Module attached and remove the cable. Take note of the orientation of the silver connectors on the cable you removed, and make sure your adapter cable is the same way round when you insert it. Put the other, smaller side into your Pi Zero, with the silver connectors facing towards the board.



A very simple Fritzing diagram just to make sure you know how to wire up the sensor

## >STEP-02

### Wire up the circuit

The circuit for this is fairly simple, especially as the PIR doesn't need a resistor as part of its setup. The PIR comes with three connections: VCC, GND, and OUT. VCC needs to be connected to a 5V power pin, GND needs to go to a ground pin, and then there's the OUT wire which will be our input. We're connecting it to pin 8, also known as GPIO 14.

If your Pi Zero has GPIO pins attached, you can do this via a breadboard; otherwise you can loop the wire around the pin holes and use a bit of Blu-Tack to keep them in place, or a dab of glue from a glue gun on a low setting. Soldering is also an option.

## >STEP-03

### Get the code

Write up or download our code for this project. This code uses two libraries: GPIO Zero and the standard picamera library. GPIO Zero can be used to get a reading from the PIR motion sensor very easily, which can then be tied into the picamera code so it takes a photo when motion is detected. We make sure the photo is given the time so you know when it was taken, and it then waits five seconds before seeing if it should take another photo. Save it as **spy.py** in the default home folder.

## >STEP-04

### Final preparations

You can run it first to give it a test. You might want to change the sensitivity, which you can do by adjusting

You should probably hide yours somewhere slightly better – its range is far, though
Camouflaged against Yoshi, the camera will take candid snaps of anyone who comes close to your game collection

the little orange screws on the side of the PIR board. Once that's done, it's best to make it so that when the Raspberry Pi boots up, it logs directly into command-line mode. This means it will use up a little less power so your battery lasts longer. It's then a good idea to open up the terminal and edit the profile config file with **sudo nano /etc/profile**. To the bottom of the file, add this line:

```
sudo python spy.py
```

> Now you need to find a good place to hide your camera

## >STEP-05

### Hide your camera

Now you need to find a good place to hide your camera. The cable for the camera is limited by length, while the PIR can have its wires lengthened, so keep that in mind when building your system. Hiding the Pi and battery behind a plush toy or photo frame can work well; you could even put a dummy photo up and cut a hole in it for the camera to look through. The PIR has quite a wide range, so put it up high where people are unlikely to look.

## spy.py

**Language**
>PYTHON 3

**DOWNLOAD:**
magpi.cc/PiSpy

```
#!/usr/bin/env python

from gpiozero import MotionSensor
from picamera import PiCamera
from datetime import datetime
from time import sleep

sensor = MotionSensor(14)
camera = PiCamera()

while True:
    sensor.wait_for_motion()
    filename = datetime.now().strftime("%H.%M.%S_%Y-%m-%d.jpg")
    camera.capture(filename)
    sleep(5)
```

## >STEP-06

### Check for intruders

All you need to do now is plug in the battery and the Pi Zero will turn on and run the script. Do some tests to make sure the camera is facing the right way. Leave it running during the day and then when you get back, plug it into a monitor, stop the script, and run **startx** to get the GUI up. From here you can see the pictures it has taken – crucial evidence to catch your dog or sibling red-handed.

**SEAN MCMANUS**

Sean McManus is the author of *Raspberry Pi For Dummies* (with Mike Cook), and *Scratch Programming in Easy Steps*. **sean.co.uk**

There are some great fantasy sprites included in Scratch, including this purple hedgehog-like creature!

Move the trampoline left and right to catch the hedgehog and bounce it back up in the air.

# BOUNCY HEDGEHOG

Spike the hedgehog loves playing on the trampoline, but he's a bit clumsy. Can you move the trampoline to stop him landing with a bump?

Who doesn't love hedgehogs? Here we are going to make a simple Scratch game with an additional dose of prickly goodness, in which you use the cursor keys to move the trampoline left and right to catch a bouncing hedgehog target. This tutorial shows you how to bring in new sprites and backgrounds, and how to use the bracket blocks and diamond blocks in your projects. You'll find these skills useful as you build other things in the future. Start a new Scratch project, and get ready to bounce!

## >STEP-01

### Prepare your artwork

For this Scratch project, you don't need the cat sprite, so right-click it in the Sprite List and then choose Delete. To add a new sprite, click the icon above the Sprite List that shows a folder and a star. Add the trampoline sprite from the Things folder, then the fantasy11 sprite from the Fantasy folder. Let's change the background too: click the Stage in the Sprite List and you will see that the Costumes tab changes to a Backgrounds tab. Click the tab and use the Import button to bring in your choice of background. For this project, we have chosen to use the atom-playground image from the Outdoors folder.

.01

when clicked
set size to 20 %
go to x: 0 y: -120
when left arrow key pressed
change x by -10
when right arrow key pressed
change x by 10

## >STEP-02

### Adding player controls

.02

Click on the trampoline (which should be Sprite1) in the Sprite List to select it, and then click the Scripts tab above the Blocks Palette. **Listing 1** shows the scripts you need to add to this sprite. Work your way down them, dragging the blocks into the Scripts Area one at a time and joining them up. Click the white holes in the blocks and type the right numbers in. Remember that the colours are a clue: to find the yellow blocks, click the yellow Control button above the Blocks Palette first.

.03

## >STEP-03

### Set up the hedgehog

Click Sprite2 (the hedgehog) in the Sprite List. Add the script shown in **Listing 2** to it. This puts the sprite at the top left of the Stage when the game begins, and gives the player a chance to spot it before it moves.

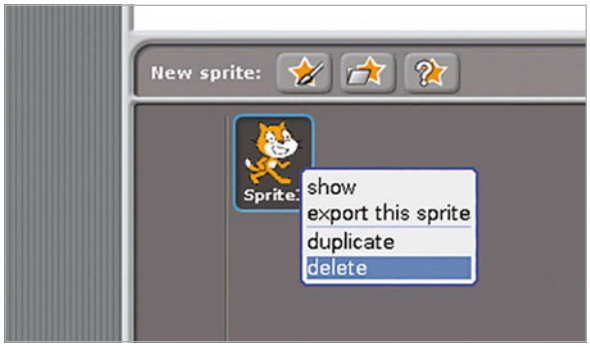
.04

when clicked
set size to 40 %
point in direction 15
go to x: -200 y: 150
wait 1 secs
repeat until y position < -120
move 6 steps
if on edge, bounce

## >STEP-04

### Add a repeat loop

We're now going to extend that script by adding some more blocks at the bottom of it. **Listing 3** shows the entire script, including the bits you've already done. Click the Control button above the Blocks Palette. Drag a **`repeat until`** block into the Scripts Area and join it to your script so far. (Make sure you don't use the repeat block with a number in it). Next, you need to drop a **`<`** block into the diamond-shaped hole. Click the Operators button above the Blocks Palette to find it. Type **`-120`** into the box on the right. Finally, click the Motion button and drag the **`y position`** block into the left box. Now, whatever we put inside the **`repeat until`** bracket will be repeated until the sprite's y position (how far up or down the screen it is) is less than -120. In our game, that means it has missed the trampoline and hit the floor.




**Above** Right-click the sprite in the Sprite List to delete it (note that the buttons to add a sprite are found above the cat)

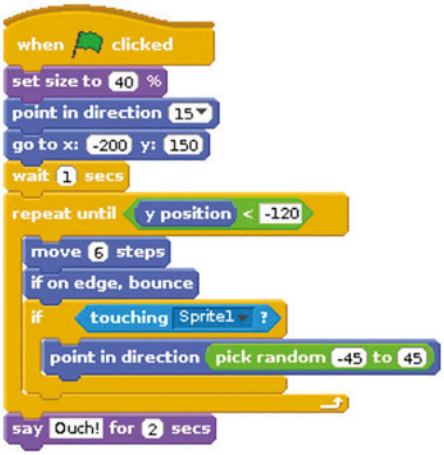
**Above** The Operators blocks include the block for picking random numbers, and the blocks for comparing numbers

## >STEP-05

### Make the hedgehog move

To make the sprite move, add the two Motion blocks shown in **Listing 4** into the **`repeat until`** block in your script. Click the green flag above the Stage to test it so far. You should see the hedgehog go to the top left of the Stage, plummet down, and stop when it reaches the bottom.

.05


when clicked
set size to 40 %
point in direction 15
go to x: -200 y: 150
wait 1 secs
repeat until y position < -120
move 6 steps
if on edge, bounce
if touching Sprite1 ?
point in direction pick random -45 to 45
say Ouch! for 2 secs

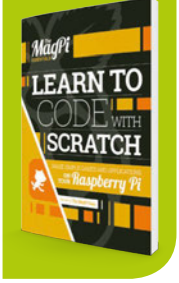
## >STEP-06

### Make the trampoline bouncy

We need to make the hedgehog bounce back up again if it touches the trampoline. Click the Control button and drag an **`if`** block into your script. Be careful with where you put it: it belongs inside your **`repeat until`** bracket, as shown in **Listing 5**. Click the Sensing button and drag in a **`touching`** block for the diamond-shaped hole in your **`if`** block. Click the menu in the **`touching`** block to choose Sprite1 (the trampoline). Inside the bracket of your **`if`** block, add a **`point in direction 90`** Motion block. Instead of putting a number in its hole, this time we'll use a **`pick random`** block with values of -45 and 45. You'll find it in the Operators section of the Blocks Palette. Now the sprite will point in a random upward direction (between 45 degrees left and 45 degrees right) if it touches the trampoline. Finally, add a **`say`** block at the end of your script, outside all the brackets. This message is shown when the game ends.

### SCRATCH ESSENTIALS

Learn more about Scratch in our book *Learn to Code with Scratch*, which takes you from beginner to pro. Find out more here: **magpi.cc/Scratch-book**



# FREQUENTLY ASKED QUESTIONS

Your technical hardware and software problems solved...

**NEED A PROBLEM SOLVED?**

Email **magpi@raspberrypi.org** or find us on **raspberrypi.org/forums** to feature in a future issue.

# BOOT DEVICES

## WHAT CAN THE RASPBERRY PI BOOT FROM?

### SD card

A micro SD card is the main boot option for the Raspberry Pi, and the easiest to use. Images can be burnt to the card or loaded via NOOBS. You can then use the card for storage if it's big enough, or make use of external storage options.

### USB

The space on SD cards is a little more limited than what you can get on USB storage, flash or otherwise; so depending on how much data you're using, it might be a good idea to create a USB-booting Raspberry Pi.

### Ethernet

The Raspberry Pi 3 added the ability for the Pi to be booted over the network. This allows you to set up a boot option for a Pi and then just plug Pis in as you see fit. It's great for businesses and enterprise solutions.

**Right** You don't have to just boot from an SD card any more

## HOW DO I BOOT FROM USB?

### Enable USB boot

You'll need to boot normally via the SD card first to enable USB booting. You'll need to edit the way Raspbian boots, so that you can then make the change to the Raspberry Pi itself before attempting to boot from USB.

### Prepare USB storage

The USB storage needs to have Raspbian copied directly to it from the SD card. You'll need to do some partitioning magic on it to get it working properly, and it will probably work a bit better if you have access to a Linux machine.

### Full instructions

The steps to create the USB-booting Raspberry Pi and appropriate USB media are a bit long for these pages, but you can read up on the full procedure on the Raspberry Pi website here: **magpi.cc/2gaEyYt**.

## HOW DO I BOOT OVER THE NETWORK?

### Debug boot mode

The initial preparation procedure is similar to booting over USB, albeit with a twist to make sure it tries to boot from the Ethernet port instead. You can start to capture data from the Ethernet after the change.

### Configuration

A few files and options need to be changed for it to all work; this includes the DHCP request and reply, along with TFTP. It's not massively complex, but you need to make sure you're doing it right for it to work.

### Full instructions

Ethernet booting is a lot more complex than USB booting, but there's great info on it on the Raspberry Pi website that can help you to get it working: **magpi.cc/2gTnVyR**. You can also check out *The MagPi* #50 for a full guide on both.

# FROM THE RASPBERRY PI FAQ
# RASPBERRYPI.ORG/HELP

**Where is the on/off switch on the Raspberry Pi?**
There is no on/off switch! To switch on, just plug it in. To switch off, if you're in the graphical environment, first exit to the Bash prompt or open the Terminal. From the Bash prompt or Terminal, you can shut down the Raspberry Pi by entering the command, `sudo halt -h`. Wait until all the LEDs except the power LED are off, then wait an additional second to make sure the SD card can finish its wear levelling tasks and write actions. You can now safely unplug the Raspberry Pi. Failure to properly shut down the Raspberry Pi may corrupt your SD card, which would mean you would have to re-image it.

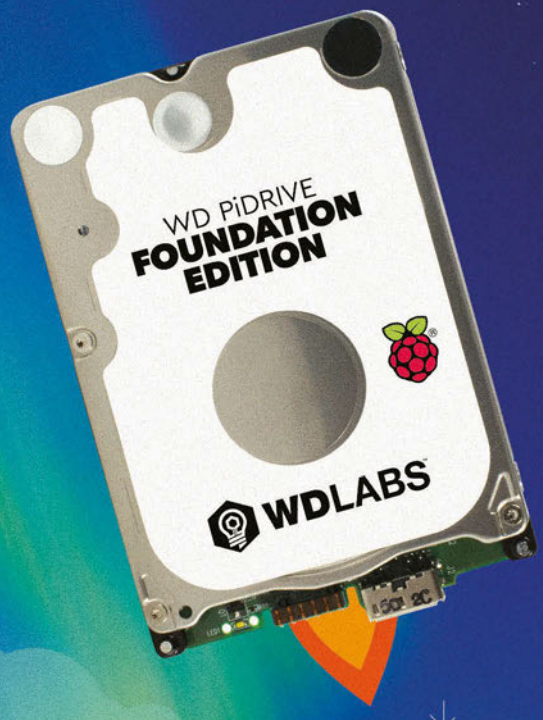
**What is NOOBS?**
NOOBS stands for New Out Of Box Software, and it's our recommended installation method. It allows you to install the distro of your choice, even if you have little to no computing or Linux experience.

**When will the next model of the Raspberry Pi be released?**
As of February 2016, the third generation of the Model B Raspberry Pi has been released. Beyond this revision, which upgraded the main processor on the board to a 64-bit version, there are no immediate plans to release any more new models. We concentrate our engineering effort on making the software that runs on the Raspberry Pi faster and better all the time, which is why you should always ensure that you are running the most recent firmware.

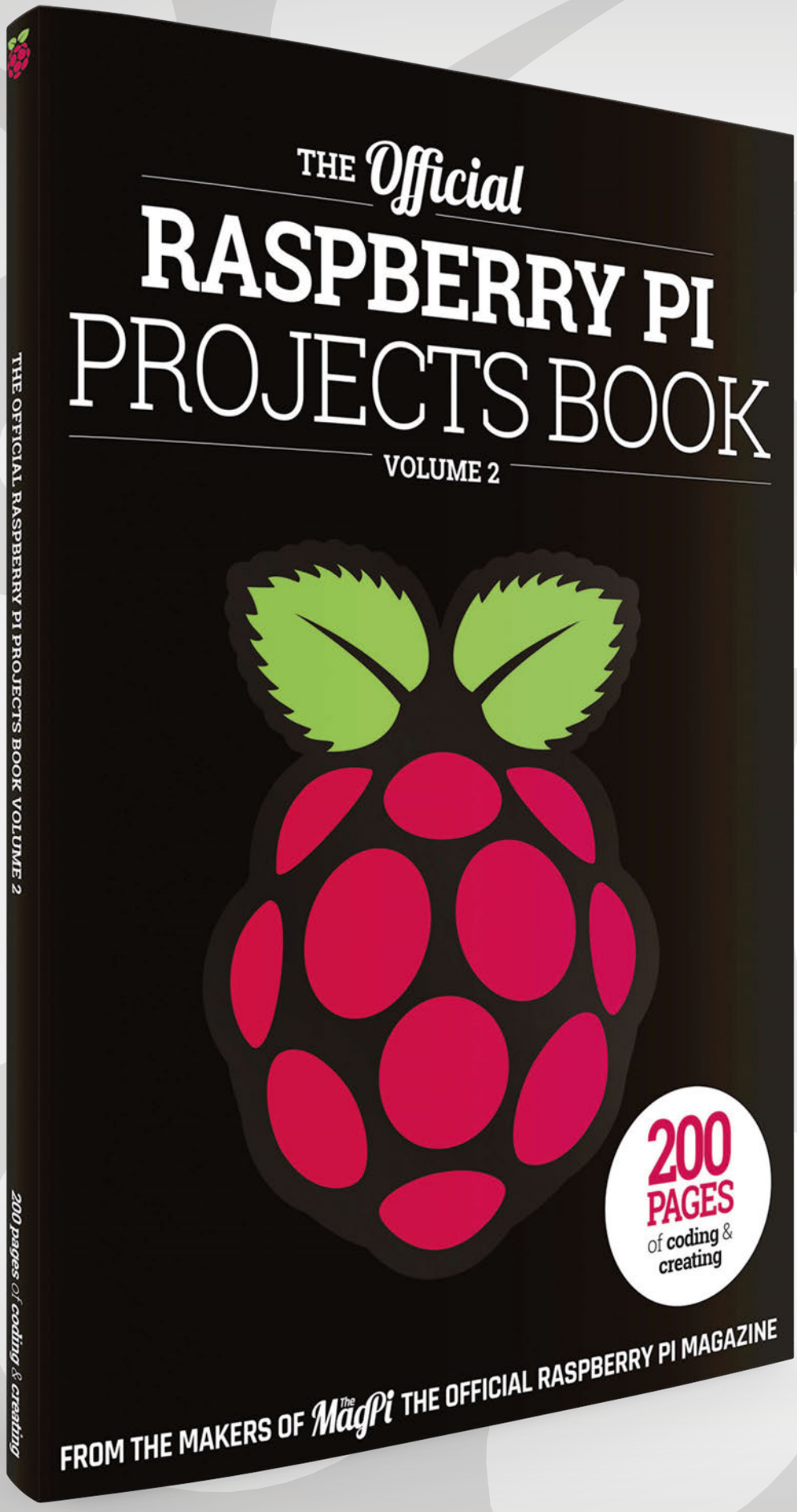
**What are the dimensions of the Raspberry Pi?**
The Raspberry Pi measures 85.60 × 56 × 21mm (or roughly 3.37 × 2.21 × 0.83 inches), with a little overlap for the SD card and connectors which project over the edges. It weighs 45g. The Pi Zero measures 65 × 30 × 5.4mm (or roughly 2.56 × 1.18 × 0.20 inches).

WD PIDRIVE FOUNDATION EDITION FOR RASPBERRY PI

Offering USB functionality and an SD card preloaded with custom starter software, the WD PiDrive Foundation Edition is your ideal storage solution for Raspberry Pi. Start creating projects in a matter of moments.




WDLABS wdlabs.wd.com/products


THE Official
RASPBERRY PI
PROJECTS BOOK
VOLUME 2
200
PAGES
of coding &
creating
FROM THE MAKERS OF The MagPi THE OFFICIAL RASPBERRY PI MAGAZINE
THE OFFICIAL RASPBERRY PI PROJECTS BOOK VOLUME 2
200 pages of coding & creating

£12.99
200 pages of
Raspberry Pi

THE Official
# RASPBERRY PI PROJECTS BOOK
VOLUME 2

**Amazing hacking and making projects**
from the creators of The MagPi magazine

**Inside:**

How to get started with Raspberry Pi

The most inspirational community projects

Essential tutorials, guides, and ideas

Expert reviews and buying advice

Available now

**magpi.cc/MagPiStore**
**plus all good newsagents and:**

WHSmith BARNES&NOBLE

Available on the App Store

GET IT ON Google Play

# RUN PIXEL ON YOUR COMPUTER

The **PIXEL** desktop is now available for Intel computers: use this month's free DVD or burn a USB drive and get the best desktop interface on your laptop

The Raspberry Pi is by far our favourite computer. So we were delighted to discover that the Raspberry Pi Foundation was making a version of Debian with PIXEL for Intel machines.

With this month's physical version of *The MagPi*, you'll find a free DVD. Load this up on your laptop or desktop and you'll find it's just like using a Raspberry Pi.

If you don't have a Debian with PIXEL DVD, don't worry. The Raspberry Pi Foundation has made Debian with PIXEL available as a downloadable ISO file.

You can use the ISO file to create your own Debian with PIXEL DVD, or to build yourself a bootable USB drive to run the PIXEL desktop on your computer.

In this feature, we'll show you how to boot your computer into Debian with PIXEL and what you can do with the OS on your home computer. We'll also look at saving files and creating a USB drive that you can use to boot into PIXEL on any computer.

### USB DRIVE

We'll show you how to create a USB drive of Debian with PIXEL for your computer. The USB drive is faster than the DVD, and you can save your work directly to it.

## THE PIXEL DESKTOP

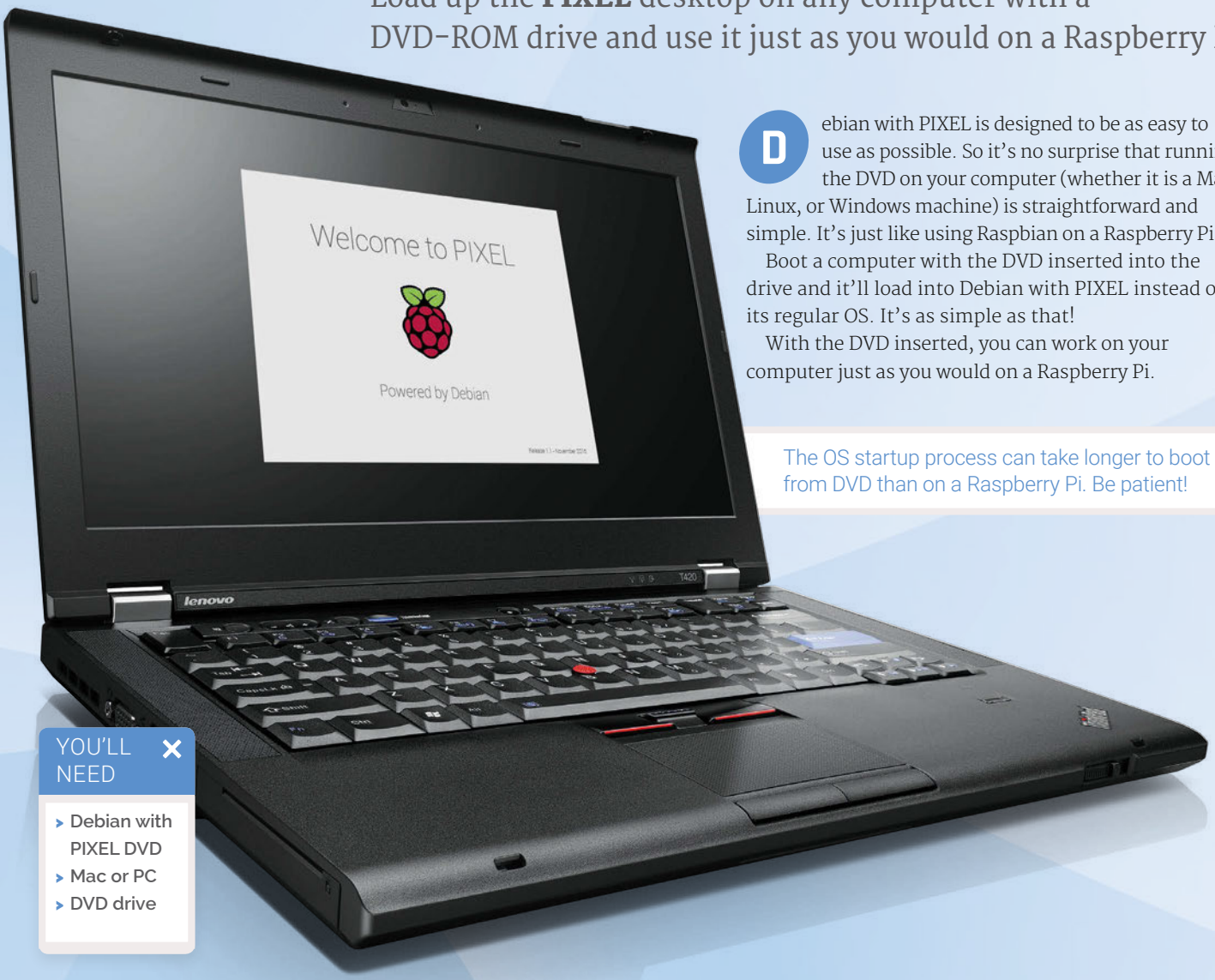
The PIXEL desktop runs on your computer just like on a Raspberry Pi. It sits on top of Debian, the same Linux OS that is compiled for ARM to create Raspbian.

## DVD DRIVE

Load Debian with PIXEL using the DVD-ROM. It runs straight from the DVD and won't interfere with your primary operating system. Remove the DVD and restart your computer when you've finished. You'll restart in your regular operating system.

# START A COMPUTER WITH YOUR FREE DVD

Load up the **PIXEL** desktop on any computer with a DVD-ROM drive and use it just as you would on a Raspberry Pi




Debian with PIXEL is designed to be as easy to use as possible. So it's no surprise that running the DVD on your computer (whether it is a Mac, Linux, or Windows machine) is straightforward and simple. It's just like using Raspbian on a Raspberry Pi.

Boot a computer with the DVD inserted into the drive and it'll load into Debian with PIXEL instead of its regular OS. It's as simple as that!

With the DVD inserted, you can work on your computer just as you would on a Raspberry Pi.

The OS startup process can take longer to boot from DVD than on a Raspberry Pi. Be patient!

## YOU'LL NEED

- Debian with PIXEL DVD
- Mac or PC
- DVD drive

## Start up

Turn on your computer and insert the DVD into the drive. Needless to say, you'll have to be using a computer with a DVD drive. If your computer hasn't got one, then you'll need to create a bootable USB flash drive to run Debian with PIXEL – see page 76.

If you are using PC-based hardware, it doesn't matter if you typically boot into Linux or Windows. Most PC computers are set up to boot from the DVD drive before the hard drive. You should hear the DVD spin and see 'Welcome to PIXEL powered by Debian' instead of your usual operating system.

If you're booting from an Apple Mac computer, then you can either hold down the **C** key on your keyboard when you hear the Apple 'bong' noise, or hold down **ALT** and choose the DVD as the startup disk.

## The boot process

After a short while you'll see a screen displaying 'Debian (Jessie) with PIXEL i386'. You will also see 'Press Esc for options' and a timer counting down.

### BOOT MENU

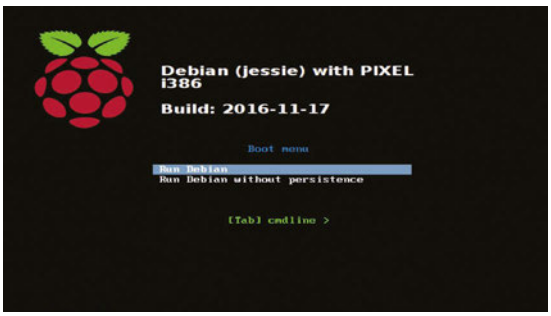
Pressing **ESC** while booting into Debian with PIXEL brings up the Boot menu options. Here you'll see two options: 'Run Debian' and 'Run Debian without Persistence'.

If you don't press **ESC**, it will automatically pick the first option. Persistence is something we'll look at a lot in this feature. It enables you to save changes to the OS while working from the DVD or a bootable USB drive.

Picking Debian without Persistence may help you boot into Debian, although you won't be able to save any files or changes you make to the OS.

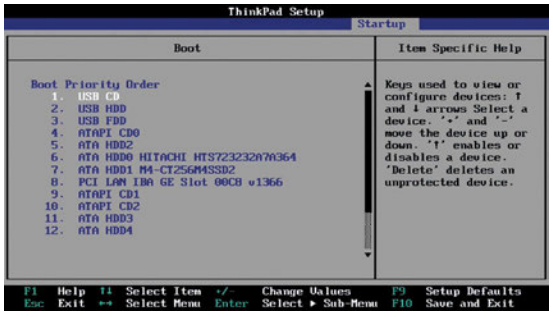
Press **TAB** to view the command-line boot for the option you have selected. Here you can add and remove specific options. Adding `nomodeset` after 686-pae can help some older Macs boot.

When you're happy with the boot option, press **ENTER** to boot the DVD.




### PC BIOS

We found Debian with PIXEL to be reliable on our test PCs. If it doesn't start, you probably need to adjust your BIOS (Basic Input/Output System) settings. Press **F1** on your keyboard (or follow the on-screen instructions) after starting up, to access the BIOS settings. These are the default hardware options for your computer (the settings that don't depend on the operating system). Use the arrow keys to move around and look for Startup and Boot. On our ThinkPad, it is called Boot Priority Order. Use the + and - keys to move the DVD drive to the first position. Choose Save and Exit, then Yes to reboot and try again.


ThinkPad Setup
Startup
Boot
Item Specific Help
Boot Priority Order
1. USB CD
2. USB HDD
3. USB FDD
4. ATAPI CD0
5. ATA HDD2
6. ATA HDD0 HITACHI HTS723232A7A364
7. ATA HDD1 M4-CT256M4SSD2
8. PCI LAN IBA GE Slot 00C8 v1366
9. ATAPI CD1
10. ATAPI CD2
11. ATA HDD3
12. ATA HDD4
Keys used to view or configure devices: ↑ and ↓ arrows Select a device. '+' and '-' move the device up or down. 'f' enables or disables a device. 'Delete' deletes an unprotected device.
F1 Help ↑↓ Select Item +/- Change Values F9 Setup Defaults
Esc Exit ↔ Select Menu Enter Select ▸ Sub-Menu F10 Save and Exit

Don't press anything. When the counter hits zero, you'll see 'Loading, please wait'.

You need to be patient when loading from a DVD. It may take a long time to start, depending on the speed of your DVD drive. Be patient.

## Logging in

During the boot process, you will see the 'raspberrypi login' appear on the boot screen. Don't enter any text; you will be logged in automatically and taken to the PIXEL desktop interface. The default username is **pi**, and the password is **raspberry**.

> "You can work on your computer just as you would on a Raspberry Pi"

## Using Debian with PIXEL

You can now use Debian with PIXEL on your computer just as you would on a Raspberry Pi. Be aware that any files you create aren't saved when you finish. Choose **Menu > Shutdown > Shutdown** when you've finished, to switch off the operating system.

# USING THE DVD

Start using **Debian with PIXEL** on your computer

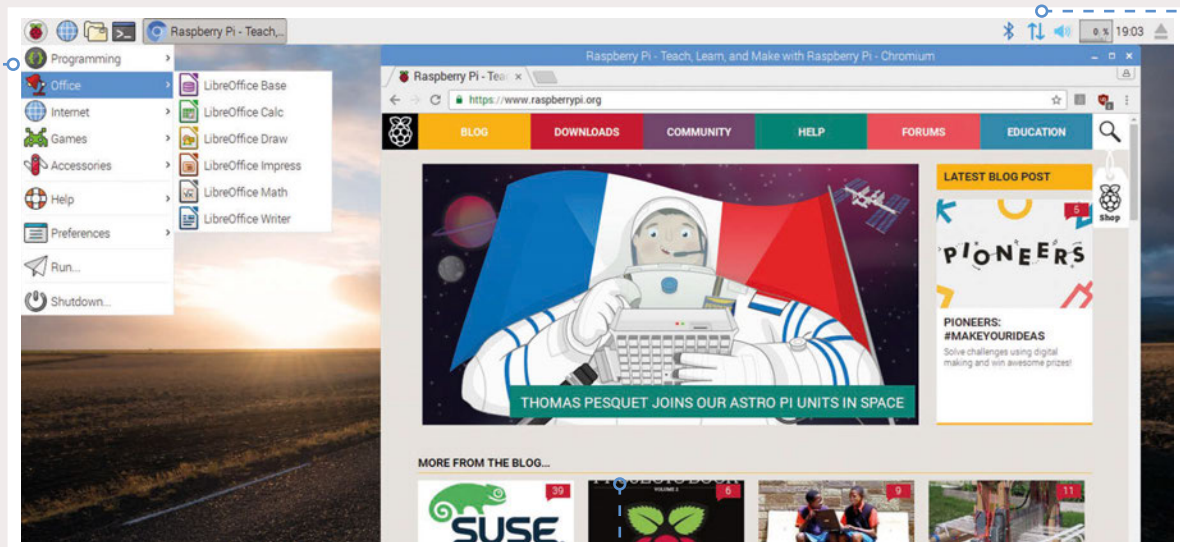
## PACKED WITH PROGRAMS

The team has curated the best office and coding software around. This software makes Debian with PIXEL an ideal creative and learning system.

## NETWORK

You can connect the OS to a local wireless network and use the built-in Chromium web browser (with Flash) to access online resources.

## THE PIXEL DESKTOP INTERFACE




## PIXEL DESKTOP

The PIXEL desktop on your regular computer looks and works much as it does on a Raspberry Pi. This makes it ideal for learning to code.

When Debian with PIXEL has loaded, you can use your computer just as you would use a physical Raspberry Pi.

Start by clicking the Menu icon in the top-left to access all the built-in software. Here you'll find the finest collection of programming tools and educational resources around. Debian with PIXEL also comes with LibreOffice, a strong alternative to Microsoft Office.

The programming tools are where it really shines. Programs like Greenfoot Java, Scratch, and Sonic Pi are all available. It even has the Sense HAT Emulator and links to all of the Raspberry Pi help and resources.

## Get online

One of the first things you'll want to do is connect Debian with PIXEL to the internet. Click on the Network icon in the menu bar to view local wireless networks. Choose a network and enter the password (the 'Pre-Shared Key'). Alternatively, if your laptop has an Ethernet socket, you can use an Ethernet cable to connect directly to a network or your home router.

Once you are online, it's possible to browse the internet using the built-in Chrome browser with Flash support. You can also download software to Debian using APT.

Programs you install using APT are stored in memory and can be utilised until you shut the computer down.

When you turn off Debian, all the files you have created (and the programs you have installed) are wiped from the system. This refresh can be useful.

Without a persistence drive, Debian with PIXEL starts up as a fresh install each time, making it handy as a teaching tool. Be careful not to lose any files, though.

## Persistence drive

If you want to be able to save files, and keep programs you've installed, you'll need a persistence drive.

The persistence drive works in tandem with your DVD. The DVD disc is used to boot the computer, and files you save into your home folder are stored on the USB drive, but everything works as a single volume.

Creating a persistence drive is pretty simple. You format a USB drive in a Linux format (typically ext4) and give it the label **persistence**.

After formatting the drive, you need to create a configuration file on the root of the drive. The file is called **persistence.conf** and contains the text: `/ union`. This configuration file tells the USB partition to work in union with the DVD.

## Format the drive

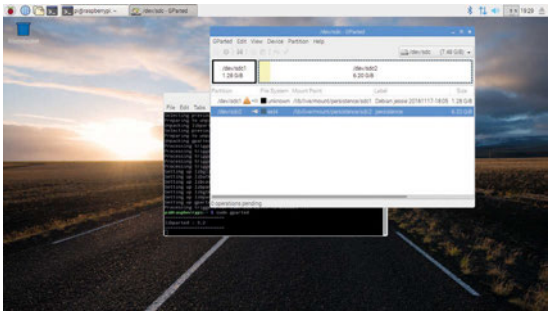
The easiest way to format a hard drive is to use an app called GParted. Open a Terminal window and enter `sudo apt install gparted`. When this is finished, enter `sudo gparted` to run the program with sudo privileges.

Click the Disks menu in the top-right of GParted (it should say /dev/sda) to view your hard drives. There will be one option for each drive on your computer.

Now attach your USB drive and choose **GParted > Refresh Devices** (or press **CTRL+R**). Click the Disks menu again and you'll see the new option.

Typically it'll have the smallest amount of space (measured in GiB). Be careful to select the USB drive and not your primary hard drive.

### PERSISTENCE DRIVE



Install GParted to format a USB drive and create a persistence drive. This drive is used to save files and changes you make to the Debian OS.

### THE STORY BEHIND DEBIAN WITH PIXEL

We caught up with Eben Upton, co-founder of the Raspberry Pi and CEO of Raspberry Pi Trading, to discuss the creation of Debian with PIXEL for Intel machines.

"I guess the real questions is 'why didn't we do it earlier?'," Eben tells us. "People used to say to us things like: 'why aren't you doing software?' and 'everyone's got a PC'."

The answer back then was: 'everyone doesn't have a PC' and 'people are scared to break them, so they don't want to get their hands dirty'.

So the Raspberry Pi Foundation concentrated its efforts on making hardware for a long time and the software for the Raspberry Pi was generic for a large part of that time. "Then Simon [Long] joined us," says Eben.

Simon has done an enormous amount of work to transform the stock Raspbian desktop into the much more polished PIXEL experience.

One thing that's superb is "the curation," explains Eben. Software like Flash, Chromium, Java, and LibreOffice requires a lot of effort to get on board. "It's getting close to being the perfect environment for productivity and learning to code.

"We've now got a piece of software that's pretty interesting. The question became: 'why are we forcing people to buy a Raspberry Pi to run it?'

"So we've come full circle," Eben concludes. "We're giving it a go to see if people like it."

First, right-click the long horizontal bar and choose Unmount (if it is available). Next, right-click on each partition in the list and choose Delete until you see a grey horizontal bar with 'unallocated'.

Choose **Partition > New**. Enter **persistence** in the Label field and click OK.

Finally, click the green tick icon (Apply All Operations) and click Apply in the alert window.

## Configure the drive

Now we need to add a **persistence.conf** file to the root of our formatted disk. Remove and reinsert the USB drive.

Enter `cd /media/pi/` and `sudo chown pi persistence` to change ownership of the drive.

Next, enter `cd persistence/` to move to the root of the USB drive, and then `sudo echo / union > persistence.conf`.

The echo command writes '/ union' to a file called **persistence.conf** on the root of your hard drive. You can check it using `cat persistence.conf`.

Now all you need to do is restart your computer. Enter `sudo reboot`. The computer will boot from the DVD drive, and use the persistence USB drive.

# MAKE A DEBIAN WITH PIXEL USB DRIVE

**YOU'LL NEED**

- USB drive
- Debian with PIXEL ISO file

What if you haven't got a DVD drive? Don't worry: you can create a bootable USB drive...

Many newer computers don't come with DVD-ROM drives. So what happens if you want to run Debian with PIXEL on your computer, but you don't have a drive?

The answer is pretty simple: create a bootable USB flash drive from the Debian with PIXEL ISO file.

Most modern computers can boot directly from a USB drive. This drive is faster and slightly more reliable than using a DVD (and older laptops can have flaky DVD-ROM drives).

You can partition a USB drive to contain both Debian with PIXEL and the persistence drive. It's like having a Raspberry Pi in your pocket that you can boot on virtually any other computer.

Sounds fun? Let's make one!

## FIX A MISSING MENU BAR

Occasionally we found the PIXEL menu bar didn't appear. In this case press **CTRL+ALT+F1** to switch to another session (enter **pi** and **raspberry** to log in). Now enter `sudo pkill lxsession` to restart the LXDE environment. You'll be taken back to the PIXEL interface with a login screen.

## Getting the ISO

You'll need the Debian with PIXEL ISO file. You can turn the DVD-ROM into an ISO file using a program called InfraRecorder in Windows (**magpi.cc/2fP6wZT**), Disk Utility in Mac OS X, or by using the **dd** command in Linux. However, it's better to download the latest version of the ISO from the Raspberry Pi Foundation's website (**magpi.cc/1MYYTMo**).

## Using Etcher

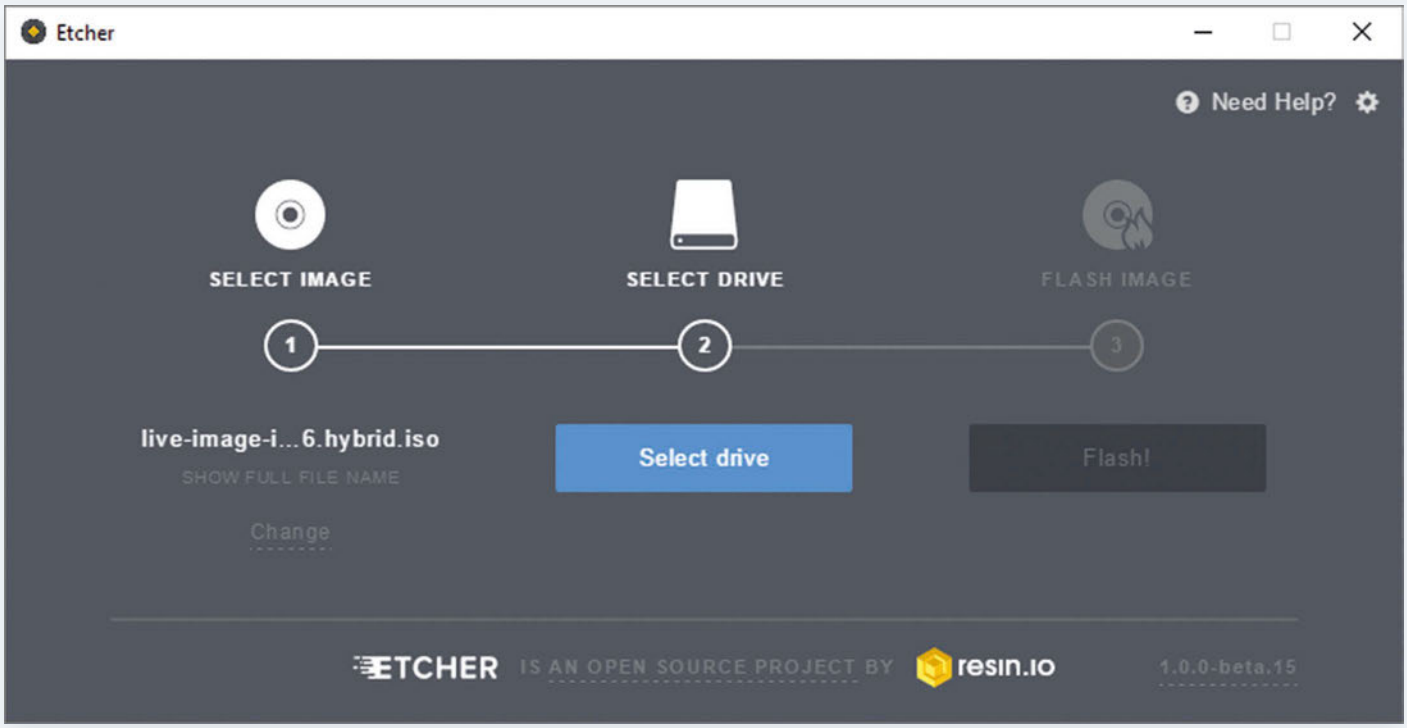
There are many ways to turn the ISO file into a bootable drive. We find it easiest to use the Etcher tool (**etcher.io**). We looked at Etcher in-depth in issue 50 of *The MagPi* (**magpi.cc/Back-issues**).

We generally use Etcher to create SD cards to boot up our Raspberry Pi, but it turns out to be just as handy for burning bootable USB drives.

Open Etcher and click Select Image. Choose the ISO file containing the Debian operating system. Now insert your USB drive. It may appear automatically under Select Drive (otherwise click Select Drive and choose the correct disk). Click Flash! to copy the ISO file to the USB drive. Enter your password, if it's requested, and click OK.

## Starting up

Now restart your computer with the USB drive attached. If you have been using the DVD, don't




forget to remove it from the drive first, or you'll boot from that instead of the USB drive.

Many PCs are set up to boot from a USB drive before the primary hard disk, but if yours is not, you may need to access the BIOS settings and change the boot order (press **F1** or follow the on-screen instructions during startup).

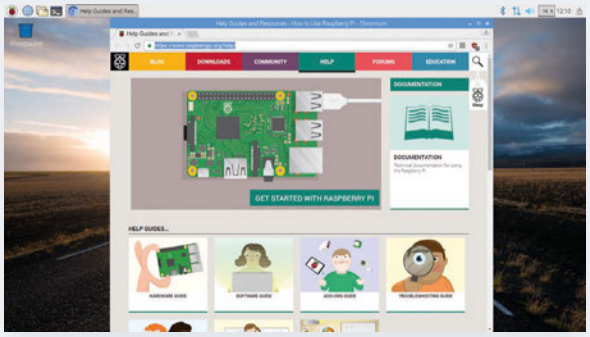
## Create persistence

If you installed Debian with PIXEL using an ISO file you created from the DVD, you will need to turn the remaining space on the USB drive into a persistence partition. You won't need to do this if you installed Debian with PIXEL using an ISO downloaded from the Raspberry Pi Foundation website (it includes a script that automatically creates the persistence partition).

Open a Terminal window and enter **`sudo apt install gparted`**. When it's finished, enter **`sudo gparted`** to run the program with sudo privileges.

### USB DRIVES FOR MAC

If you're struggling to get the USB drive to start up on a Mac, try formatting the drive into two partitions. The first partition should be FAT32 and 1.5GB in size with a boot label. The rest of the drive should be a Linux ext4 partition with the label **persistence**. You then use 7z to unzip the contents of the ISO into the FAT32 partition. We found these instructions on CosmoLinux helpful: **magpi.cc/2ggdzrs**



**Above** Use Etcher to record the ISO file into a bootable Debian USB drive for your computer

**Left** With your USB drive, you can boot directly into Debian with PIXEL: it's just like using a Raspberry Pi computer

Click on the Disks menu and choose the USB drive from your disks. It should be the smallest drive.

Right-click on the grey horizontal bar marked 'unallocated' and choose New.

Enter **persistence** in the Label field and click Add. Click the green Apply icon and click Apply in the Apply Operations to Device alert window. Click Close when you see 'All operations successfully completed.'

Restart with **`sudo reboot`**. This will mount both the boot volume and the persistence volume.

Open a Terminal window and enter **`cd /media/pi.`** Here you will see the mounted persistence partition (enter **`ls -l`** to view the volume and its permissions). You need to take ownership of it to make changes. Enter **`sudo chown pi persistence`** to take change the ownership of the drive from root to pi. Enter **`ls -l`** to view the permissions again.

Use **`cd persistence/`** to move to the root of the drive. Enter **`echo / union > persistence.conf`** to create the persistence configuration file. Use **`cat persistence.conf`** to check the contents of the configuration file. It should be just '/ union'.

Enter **`sudo reboot`** one last time. You will now boot into Debian with PIXEL from your USB drive.

## Maker Says

"It's easy to build and lots of fun to use"

**Mime Industries**

# MIROBOT V2 PROGRAMMABLE ROBOT

Platform-agnostic and tool-free, could the Mirobot be the perfect first robot for budding makers?

When a Mirobot drops through the letter box, it comes as a real surprise. Whereas most robots come in bulky packaging, the Mirobot sits in a simple cardboard box the size of a few stacked CDs. It's easy to assume that it's small as a result, but it's actually testament to creator Ben Pirt's talents: the packaging for the Mirobot becomes the robot itself.

The Mirobot's body is constructed from laser-cut wood, and the kit takes the form of sheets of wood with the pieces still in place. Holes are cut out for the non-wood components, like the motors and circuitry, and the whole lot carefully assembled and held together with two elastic bands at the corners.

The result: very little goes to waste. The outer framework from each slab of wooden components is useless once popped out, but can be disposed of as compost or used as fuel in a fire, while everything else bar the cardboard outer gets put to use, even the elastic bands.

The build process is tool-free and simple: someone with no experience of electronics can easily follow the instructions, printed in colour with illustrative photographs, and end up with a working Mirobot in around 10 or 20 minutes. The only slightly tricky steps are inserting the non-keyed wires into the main circuit board, which must go in a specific way around, and calibrating the pen using the included tool, laser-cut from wood, as with everything else in the kit.

## Ease of use

Once built, the Mirobot is powered by four AA batteries and operates entirely independently. Initially, the on-board wireless hardware, which uses the low-cost ESP8266 WiFi module, broadcasts as an unencrypted hotspot to which you can connect a smartphone, tablet, Raspberry Pi, or laptop in order to configure the device.

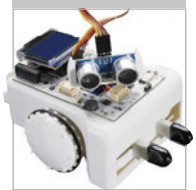
Here's the second surprise: there's no software to install. Everything on the Mirobot, from upgrading its firmware to programming it, takes place in your web browser. You can choose to use it offline in unencrypted or encrypted hotspot mode, or give it your WiFi details to connect it to your home or school network.

If you choose the latter, you gain access to the Mirobot Apps from the official website. These offer a wide range of choices for programming the robot: a spartan drag-and-drop block-based user interface which is available locally; the Blockly, Snap!, and Scratch block-based languages, though the latter is Adobe Flash-based and unavailable from a Raspberry Pi;

## Related

### ARCBOTICS SPARKI

The Sparki comes with sensors missing from the Mirobot and an on-board display, but it's considerably more expensive as a result.



£145/$149

arcbotics.com

mime.co.uk

**£60 / $75**

**Right** The full kit comes with a chassis to put your robot on

Python or JavaScript text-based languages, complete with in-browser development environment and syntax highlighting; and a point-and-click interface for easy drawing. There's also a direct remote control option, which allows you to control the Mirobot using on-screen buttons; coupled with a smartphone, this turns the Mirobot into an admittedly slow remote-controlled car. These are all available for offline use, too, through a Chrome app which can be installed into Google's web browser on all supported operating systems.

In use, the Mirobot acts like your average turtle robot. It can raise or lower the pen, which you must provide yourself, to draw shapes on paper, turn in either direction, move forwards and backwards, and emit a beeping noise from its on-board speaker. The updated kit, now in version 2.0, also includes an optional infrared sensor, which when installed can be used to turn the Mirobot into a line-following robot.

## More to learn

The manual does, however, stop just short of where an absolute beginner might have liked. In particular, you'll find no reference to fitting the line-follower in the printed documentation, the need to calibrate the pen's position and height, and calibrating the distance the robot travels by drawing and measuring a line, in order to produce clear shapes from your programs.

Fortunately, the official website includes a learning resource section with easy-to-follow guides for all these tasks. Cleverly, they're also capable of connecting to any Mirobot on your local network: the page for calibrating the motors, for example, allows you to press a button to draw a line, then type in the length once measured, the result of which is sent back to the robot and stored in its permanent memory automatically. You'll also find guides to drawing shapes, using the line-follower, and technical documentation on the Mirobot's communications protocol.

## Last word

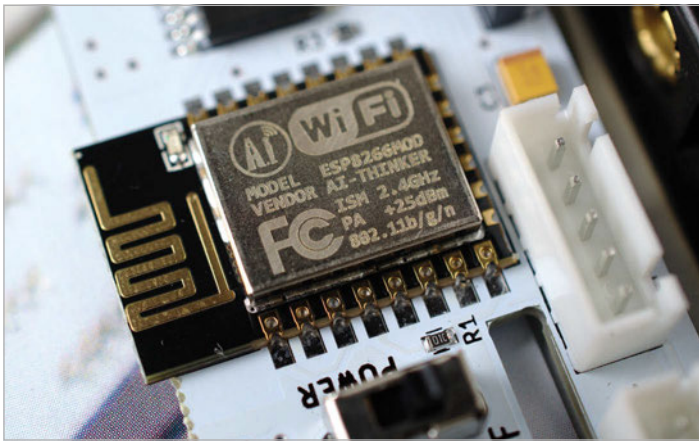
**It's impossible to dislike the Mirobot. From its simple tool-free build process and clever packaging to its browser-based user interface, the machine is a joy from the moment it arrives and a sure-fire hit with younger makers.**

★★★★★

MirobotApps
Blockly
This app allows more advanced programming techniques to be used to control Mirobot
Chrome App
Check out the standalone Chrome App which lets you run all of the apps from this site.
JS
Javascript
Program Mirobot using the full power of Javascript directly from your browser
Mirobot UI
This is the built-in simple drag-and-drop app used to program Mirobot
Miro
Point & Click
Click to control Mirobot and see the program it would use
Python
Program Mirobot using the Python language from within your browser
Remote Control
Use buttons to directly remote control your Mirobot
SCRATCH
Scratch
Use Scratch to program your Mirobot (requires Flash)
Snap!
Snap! is very similar to Scratch and is good for using the addon sensors in code

**Above** Program your robot from a web browser for maximum ease

**Below** It comes with WiFi built into the board so you can access it



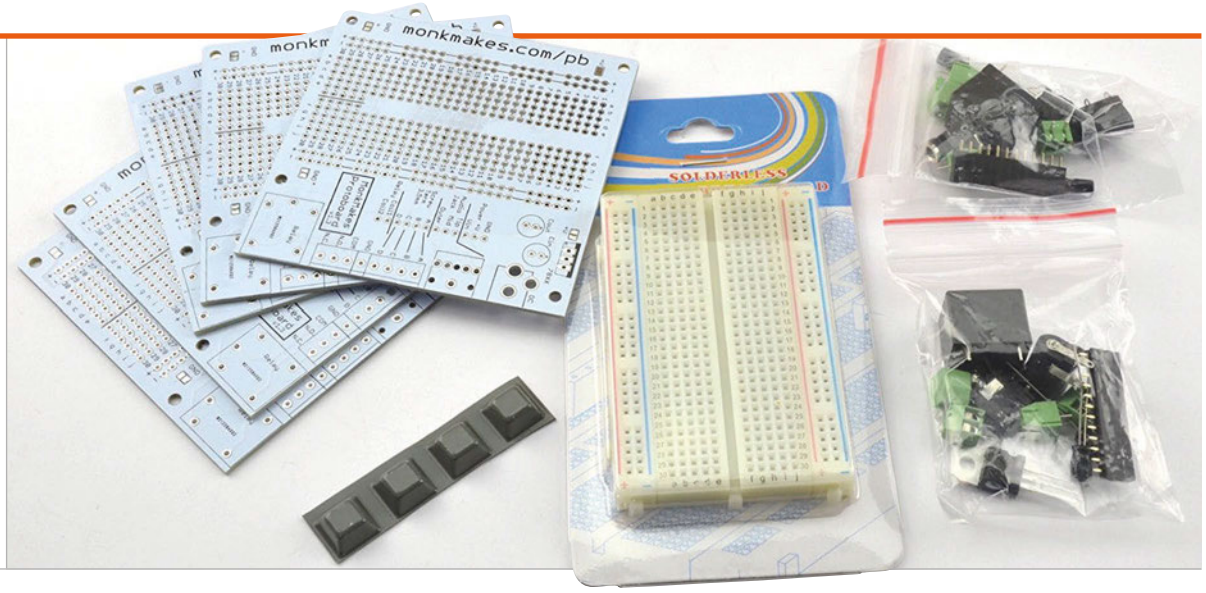
monkmakes.com/pb

**£23 / $30**

## Maker Says

"Makes the creation of one-off projects and prototypes as simple as possible"

**MonkMakes**




# PROTOBOARD

A PCB breadboard-style prototyping board with extra features

Like many electronics hobbyists, Simon Monk likes to prototype projects on a solderless breadboard before transferring them to something more permanent once everything is tested and working correctly. He says he's never been completely satisfied with other prototyping board designs, though, so he decided to address what he saw as their shortcomings. The result is the MonkMakes ProtoBoard. Following a successful Kickstarter campaign, it's now available as a deluxe kit from Amazon (**amzn.eu/3YhwMaK**).

Inspired by Adafruit's Perma-Proto boards, the ProtoBoard mirrors the layout of a standard solderless 400-point breadboard, featuring the usual interconnected rails and rows, but adding longer rows at the bottom of the board for modules that use 2×4 or 2×5 pins, such as the ESP01 or NRF24 modules: a nice touch.

The most obvious difference, however, is the addition of an area on the right-hand side where you can put extra components that don't fit straight onto a breadboard. These include a standard 'sugar cube' relay, 3.5mm audio jack, and general-purpose screw terminals. There's also an area at the top for creating a power supply by adding a barrel jack and linear voltage regulator and capacitors. This can be connected to one or more power rails by soldering the +V and GND bridges on the board.

Two packs of components are supplied in the deluxe kit available from Amazon, along with a half-size solderless breadboard and four rubber feet. When prototyping projects, Simon advises sticking the breadboard permanently to one of the ProtoBoards and adding a full set of components to the side panel. Then, once your circuit is working as intended with this setup, it should be very easy to transfer (or copy) breadboard components to another ProtoBoard placed to the side, as well as replicating any components used in the side area. It's a neat idea that really does help.

As well as being suitable for any project that you could start with a 400-point solderless breadboard, the ProtoBoard can be used for those that feature a microcontroller such as a Particle Photon or Arduino Pro Mini – you could even use it to make an off-board Arduino.

## Related

### ADAFRUIT PERMA-PROTO

Available in three sizes, this PCB prototyping board matches the design of a classic breadboard, without the extra connectors you find on the MonkMakes ProtoBoard.

**£10 / $13**

magpi.cc/2fPNZwh

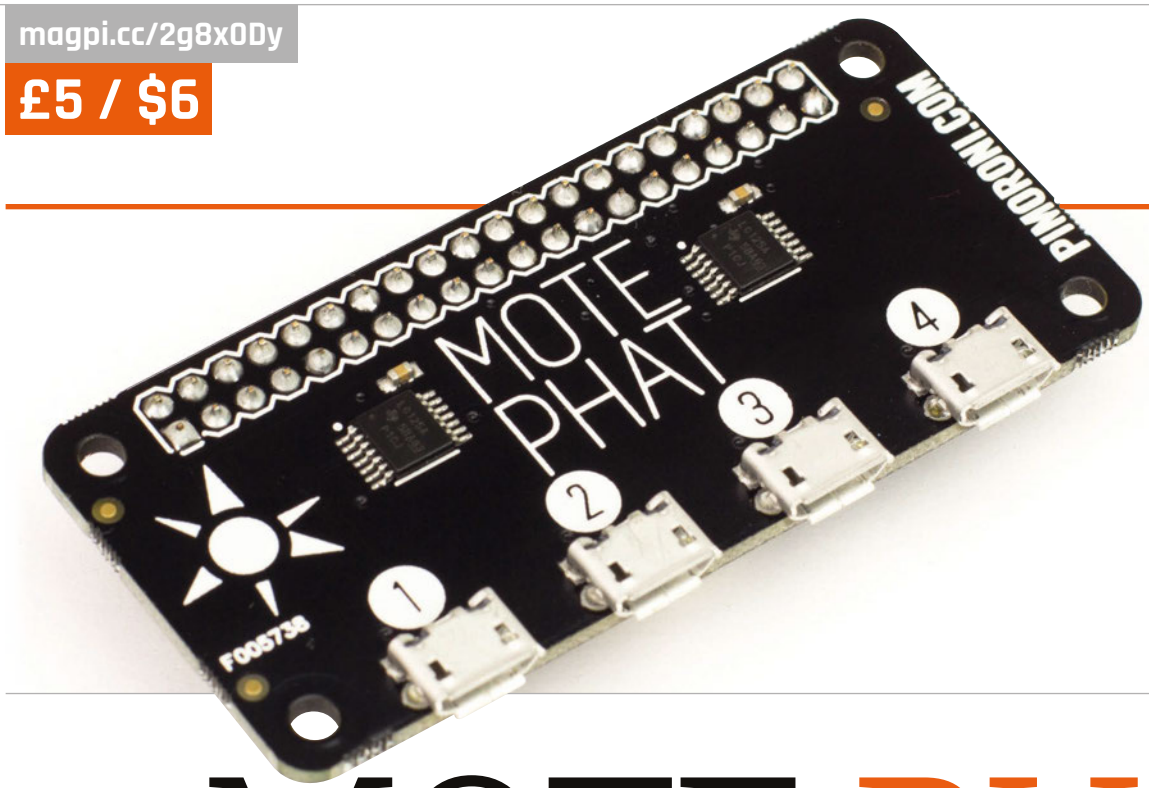
## Last word

**With a well-thought-out design, the ProtoBoard makes it easier to transfer projects from a solderless breadboard to a more permanent PCB, featuring through-plated connection holes. In particular, the side panel comes in very handy for components that don't fit straight onto a breadboard.**

★★★★☆

magpi.cc/2g8x0Dy

**£5 / $6**




## Maker Says

"Run our beautiful Mote lights straight from your Pi or Pi Zero"

**Pimoroni**

# MOTE PHAT

A more compact way to control Mote light strips

Featuring 16 super-bright RGB LEDs, Pimoroni's Mote 'stick' light strips (£8.50 each) are ideal for under-shelf/cabinet mood lighting, among other applications, and can be used to create some impressive effects. Until now, a Mote Host has been required to control the sticks, plugging into any computer – including the Pi – via USB. For a far more compact and convenient setup, the Mote pHAT does the same job. Its pHAT form-factor matches that of the Pi Zero perfectly, so you can tuck it discreetly away out of sight. You will need to solder the female GPIO header onto the Mote pHAT first, although at least this does give you the option of using a stacking header to use it alongside other add-on boards: you could get Mote sticks to display sensor readings from an Enviro pHAT, for example, or control them with buttons or switches.

As with the Mote Host, there are four output channels via micro USB ports for controlling up to four Mote sticks independently of each other. A Python library (**magpi.cc/2fw40FX**) is provided, along with a few examples to get you started. These include an impressive rainbow effect, colour cycling, and CheerLights web control via Twitter. The library itself is easy to use, enabling you to set each individual RGB pixel on each of the four channels. They are triggered with a `show` command, and `clear` is used to turn them off. Using a `for` loop, we soon managed to create a simple chase lights effect – handily, if you exceed the highest pixel number, it wraps round to the start of the strip again. Since the LEDs are APA102 standard (aka DotStar) with a fast data rate, there are none of the timing issues you might get with NeoPixels and they respond almost instantaneously.

You can even control them from your phone or from a web browser by setting up a Flask API. Pimoroni's step-by-step tutorial (**magpi.cc/2g9g143**) shows how to use HomeKit to control the lights from an iPhone, using Siri voice commands to turn them on and off and to set the colour – a good way to impress friends! The same result should be possible using Google Assistant or Alexa, too.

## Last word

**The Mote sticks are excellent, even better than NeoPixels, and the new Mote pHAT offers a compact, discreet way of controlling them without the need to connect to the USB port of a laptop or Pi. At just £5 it's great value, although of course you'll still need to buy sticks (and cables) to use with it.**

★★★★☆

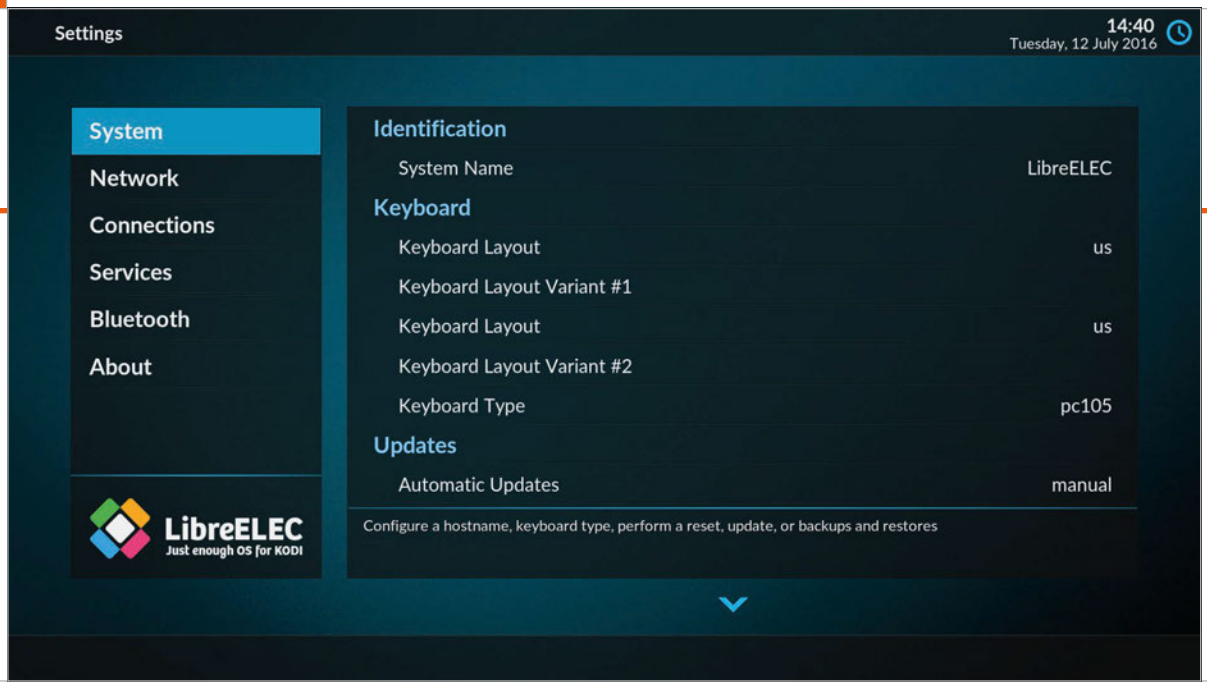
## Related

### MOTE HOST

The original Mote controller plugs into the USB port of any computer, including Pi 3, Pi 2, or B+. Like the pHAT, it features four micro USB outputs.

**£11 / $14**

magpi.cc/2g9iyLs




**Maker Says**

"Just enough OS for Kodi"

**LibreELEC**

# LIBREELEC 7.0

## A new media centre OS for the Raspberry Pi with familiar ancestry, is LibreELEC an upgrade over OpenELEC?

It's been a while since we've looked at a new media centre for the Raspberry Pi. The 'market', for lack of a better term, has been wrapped up for a little while now by OSMC and OpenELEC, so it's not like there's been much call for another one. Both are great and offer something slightly different over the other, so unless you're going to reinvent the wheel, there's not much more to add.

Which brings us to LibreELEC. It doesn't reinvent the wheel, but what it does is out-OpenELEC OpenELEC; it's probably the slimmest and most lightweight Kodi (née XBMC) OS you'll find. This comes as no surprise when you learn of LibreELEC's lineage as an offshoot of OpenELEC itself. If you know your open-source software, it's a little bit like OpenOffice and LibreOffice, albeit with no corporate buyouts involved.

LibreELEC aims to carry on the more open, free spirit of earlier OpenELEC releases, while still staying at the bleeding edge of Kodi's release schedule. There's also a commitment to stability, which is very welcome. You can see their cutting-edge credentials on the website; as Kodi ramps up to release version 17, each beta release is closely followed by a pre-release of the next LibreELEC.

For this review, we're concentrating on their stable release at the time of writing, LibreELEC 7.0. Installation is a doddle and it's handled by software developed for all major platforms (including a generic Linux one!). This software allows you download an image (stable by default, but you can select the beta), choose a destination for the image, and then burn it. It's easy to use and very fast as well, although the speed is probably down to how small the image itself is. Usually something like Raspbian can take up to ten minutes to burn to an SD card, whereas LibreELEC was done in a minute. There's even a progress bar to keep track of it. OpenELEC had a similar piece of software, but it was never quite as developed as this.

### Media consumption

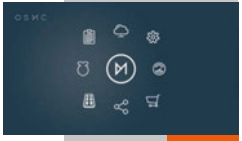
First boot is easy and familiar, too. The SD card is resized to fill the space, and a couple of other minor tasks are performed before it reboots into the media centre. The reboot countdown timer takes longer than the initial operations on a Raspberry Pi 3, and the true first boot itself is over pretty quickly.

From here things start to look a lot more familiar. LibreELEC comes with the default version of Kodi, rather than a reskinned one like OSMC. You're asked some basic details, and then it immediately allows you to connect to WiFi using the Raspberry Pi 3's on-board

**Related**

**OSMC**

OSMC dispenses with the standard Kodi interface to use its own take. This makes it flow slightly differently while still having a strong base.



FREE

osmc.tv

libreelec.tv

FREE

**Left** Welcome to Kodi, powered by LibreELEC

wireless. It also has a pretty good range of codecs for WiFi dongles if you're using a Pi 2. With that set, you're done and ready to start adding media sources. This was one of the nice things about OpenELEC: it was fast and simple, offering the Kodi environment a lot of people are already familiar with.

work absolutely fine on the Pi 2 and Pi 3. Particularly in the case of OpenELEC, unless you're experiencing issues it's not really worth the hassle of reinstalling and getting all your media set up again. While LibreELEC does also support first-generation Raspberry Pis (and the Pi Zero),

> "The way LibreELEC is built seems just that little bit more optimised for the Raspberry Pi"

Using it feels just a little bit faster and smoother as well. Since the Pi 2 release, Kodi hasn't had any issues on the Pi, but the way LibreELEC is built seems just that little bit more optimised for the Raspberry Pi. The UI flows well, media playback works absolutely fine, and it didn't break at all during testing. There's no sacrifice in terms of functionality or ease of use, either.

It's not exactly a game-changer, though. OpenELEC and OSMC you're still limited a bit by the CPU on those lower-powered models, so again there's not much of a difference in the performance.

Kodi 17 is just on the horizon, though. The software releases fairly regularly as it is, but this could make for a shift in the way the Pi media centres perform in the near future. With LibreELEC working closely with Kodi, much like OpenELEC does, we're very interested to see how this next generation of OSes will perform.

**Above** The installation tool is quick and easy to use

## Last word

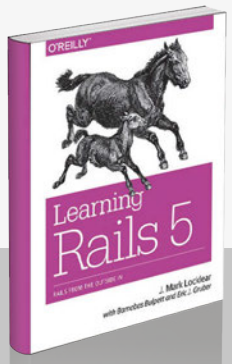
**If you like your Kodi default, clean, and lightning fast, LibreELEC is the OS for you. May not be worth the hassle of an upgrade if your current HTPC Pi is working fine, though.**

★★★★★

# RASPBERRY PI BESTSELLERS RAILS 5

**The latest Rails can do anything; probably even make your coffee!**

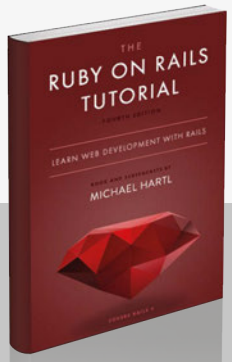
## LEARNING RAILS 5: RAILS FROM THE OUTSIDE IN




**Author:** J Mark Locklear
**Publisher:** O'Reilly
**Price:** £31.99
**ISBN:** 978-1491926192
**magpi.cc/2gFZC78**

A very good intro book, but errors in code mean you'll also learn from debugging as you go along. Sample projects build very logically through learning how to 'do' Rails.

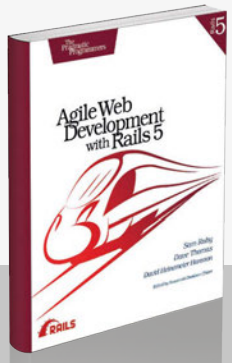
## RUBY ON RAILS TUTORIAL




**Author:** Michael Hartl
**Publisher:** Addison Wesley
**Price:** £40.69
**ISBN:** 978-0134598628
**railstutorial.org**

Hartl's comprehensive boot camp of a book (and online read) is highly praised by those who suit his immersive teaching methods. An excellent grounding, well updated for Rails 5 features.

## AGILE WEB DEVELOPMENT WITH RAILS 5

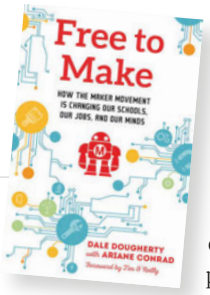



**Author:** Sam Ruby
**Publisher:** Pragmatic Bookshelf
**Price:** £32.99
**ISBN:** 978-1680501711
**magpi.cc/2gG1mNr**

A beginner's guide into which agile, Ajax, REST, testing, and many best practices are closely woven into both its warp and its weft. Clear and thoughtful updates for Rails 5.

# FREE TO MAKE

**Authors:** Dale Dougherty & Ariane Conrad
**Publisher:** North Atlantic Books
**Price:** £17.50
**ISBN:** 978-1623170745
**magpi.cc/2gFUrnw**




Few revolutionary changes in the past have been as well chronicled as the still nascent maker movement, but if anyone is well-placed to give that commentary, it's *Make:* magazine and Maker Faire founder Dale Dougherty. As well as the who, what, and where of the movement, Dougherty gives us a philosophical background to this "global countercultural phenomenon", which empowers people to move away from passive consumerism.

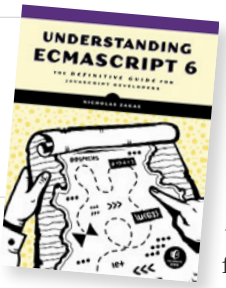
This exploration starts off with a look at the diverse range of makers – and their projects – spurred on by a love of what they are doing. The 'where' takes us on a tour of makerspaces, fab labs, and less formal workshops. The 'how' explains the boards, tools, technologies, and whole maker ecosystem. The chapter on 'The Maker Mindset' is an essential examination of what drives us to create – from playfulness, to a need to understand more fully.

The closing chapters on makers' impact on the world, and a look at future development, provide food for thought. The notes, chronology, and bibliography are also of benefit. This fascinating book should, as Tim O'Reilly puts it in the foreword, help you to "find your own inner maker. And start to measure yourself by what you make, not what you own or buy."

**Score** ★★★★☆

# UNDERSTANDING ECMASCRIPT 6

**Author:** Nicholas C Zakus
**Publisher:** No Starch
**Price:** £24.99
**ISBN:** 978-1593277574
**magpi.cc/2gFYgsZ**




ECMAScript 6 is a huge set of changes to JavaScript – everything from Promises, to improve asynchronous programming, to several improvements to the good old array. What's needed to get the average – or even occasional – JS coder up to speed is a guide to put these changes in context, and to show what they can do. Enter Zakas, creator of ESLint, and a very experienced JS developer.

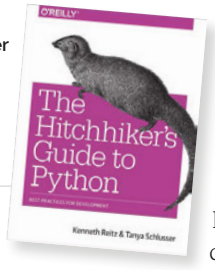
Starting with Block Bindings, one of ECMAScript 6's solutions to previous versions' problematic variable declarations, the author demonstrates admirable clarity in his explanations of how and why things work, followed by best practice examples. So in the chapter on Iterators and Generators, Zakas gives you the historic background, then changes from 5 to 6, how to implement, and only then – understanding accomplished – 6's built-in Iterators.

Classes are a big feature in 6, and get appropriate treatment – including their first class citizenship (you can pass them into functions as arguments). ECMAScript 6's tail call optimisation, changes to defining Object String Tags, user-definable non-enumerable and non-writable object properties: it's all here. Rounds off with a guide to ECMAScript 7 – or ECMAScript 2016 – first of the new, yearly release cycle, containing only minor changes. Very readable. Recommended to any JavaScript user looking to grok ES6.

**Score** ★★★★★

# THE HITCHHIKER'S GUIDE TO PYTHON

**Authors:** Kenneth Reitz & Tanya Schlusser
**Publisher:** O'Reilly
**Price:** £17.99
**ISBN:** 978-1491933176
**magpi.cc/2gFXdsS**




"Python is big. Really big" – yes, there are plenty of *Hitchhiker's Guide to the Galaxy* references here, a change at least from the obligatory Monty Python quotes in other Python works. Whimsy aside, this is a useful guide to immersing yourself in all things Pythonic, taking the confident beginner or intermediate programmer on a journey through the Python programmer's ecosystem, everything from good code writing, to reading and learning form other programmers.

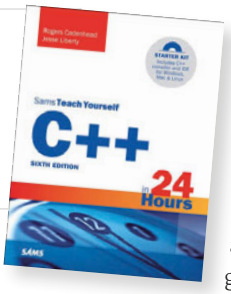
The long chapter on "reading great code" is particularly useful for those looking to step up from beginner status, but we start with a look at the programmer's environment, from Python version and implementation, through virtualenv and its alternatives, to the ever expanding choice of text editors and IDEs – where some may disagree with the authors' preference for proprietary choices.

The final section, covering libraries, features several chapters, with coverage of those for user interaction, code management, interfaces, data manipulation, and data persistence. Along the way this takes in everything from Jupyter Notebooks, through Continuous Integration, to the little known (but passionately advocated) Pyramid web framework. For anyone lacking a coding mentor, and/or a group of experienced Python using co-workers, this is a very special sort of 'missing manual'. Recommended.

**Score** ★★★★☆

# C++ IN 24 HOURS SAMS TEACH YOURSELF

**Authors:** Rogers Cadenhead & Jesse Liberty
**Publisher:** SAMS
**Price:** £21.99
**ISBN:** 978-0672337468
**magpi.cc/2gFZFzH**




C++ is full of danger for the unwary, as it doesn't provide the protections of higher-level languages like Java and Python, allowing you to shoot yourself in the foot, or even remove your leg. Nevertheless, much development on the range of interesting and low-cost boards available to makers needs C or C++ (or a subset thereof), and newbies still find themselves wanting to learn C++ to tackle embedded projects. It also remains fairly essential in gaming engines and other tasks where performance is critical.

SAMS' popular '... in 24 hours' series break down complex learning subjects into hour-long lessons to give time for digestion between sessions (you won't profit from trying to work through the book in one caffeine-fuelled 24-hour session!). The success of the format is dependent upon every lesson building steadily at a good pace, and in good order.

No intro book can teach you everything about the sprawling C++ language. The authors make some sensible choices, and this is a good step up the first few rungs of a precarious ladder. However, some updates for new language features such lambda expressions don't feel well integrated into the text. On balance, a good introduction, but you'll need more resources to get further.

**Score** ★★★★☆

# ESSENTIAL READING: NEW YEAR RESOLUTIONS

**Take the challenge: it's time once again to resolve to learn something new.**

## Resolution: Functional programming

### Functional Programming in JavaScript

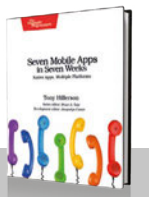
**Author:** Luis Atencio
**Publisher:** Manning
**Price:** £36.62
**ISBN:** 978-1617292828
**magpi.cc/2gFPDPb**

Learn the benefits of the functional programming paradigm using a practical language that you know and love, and an authoritative guide.

## Resolution: Code a native mobile app

### Seven Mobile Apps in Seven Weeks

**Author:** Tony Hillerson
**Publisher:** Pragmatic
**Price:** £28.50
**ISBN:** 978-1680501483
**magpi.cc/2gFYyjz**

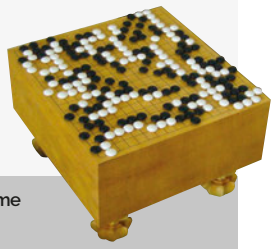



Get to grips with building mobile apps for all devices with Pragmatic's excellent and practical 'Seven Weeks...' series.

## Resolution: Learn to play Go!

### Go / Wei Qi / Baduk

**Author:** British Go Association
**Publisher:** Many online resources
**Price:** Free – download GNUGo
**ISBN:** N/A
**magpi.cc/2gFZ290**



Learn the 5,500-year-old strategy game that's taxed the finest AI researchers' minds. Play online or join a club.

## Resolution: Debug some software!

### Effective Debugging

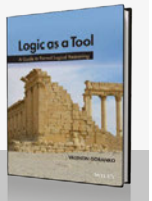
**Author:** Diomidis Spinellis
**Publisher:** Addison Wesley
**Price:** £24.99
**ISBN:** 978-0134394794
**magpi.cc/2gFVUdF**

Bugs are everywhere; learn the advanced methods, strategies, techniques, tools, and tactics to squash even the most persistent.

## Resolution: Learn to think logically!

### Logic as a Tool

**Author:** Valentin Goranko
**Publisher:** Wiley
**Price:** £45.00
**ISBN:** N/A
**magpi.cc/2h8d8kB**




Classical logic is hard but rewarding – helping with clarity in programming, business decisions, and whatever life throws at you.

# THE MONTH IN RASPBERRY PI

Everything else that happened this month in the world of Raspberry Pi

## CHRISTMAS IS ALL AROUND US

By the time you read this, you'll probably be doing last-minute preparations for the big day. Don't forget to brine that turkey! You may also already have a load of Christmas decorations up, but there have been a few extra ones we've seen over the past month that you might want to consider if you're a truly massive Raspberry Pi fan.

**magpi.cc/2gBswFv**

Raspberry Pi's Alex Bate designed these 3D-printable decorations for the Raspberry Pi and Code Club fan in all of us. They're quick to print and all you need to do is add some string to hang it up on the tree!

**magpi.cc/2hfSEtl**

We think this is very cute – a circuit board built to look like a 3D Christmas tree, with LEDs and resistors you can add to it. You then pop it on the GPIO pins and program it to do whatever you want.

**magpi.cc/2hfT3fl**

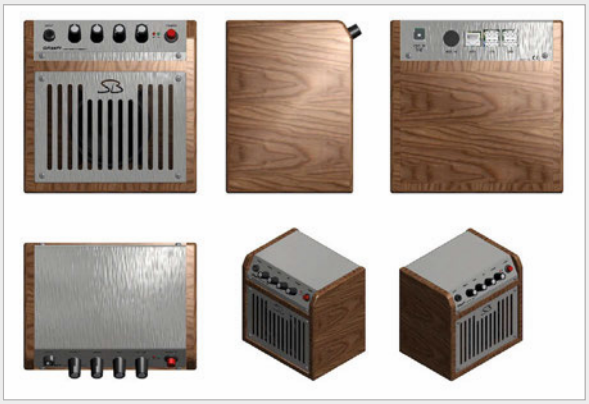
This glowing festive house kit is wonderful – use a bit of Bare Conductive's electric paint and your own creativity to make this lovely decoration. It doesn't use a Pi, but it's great nonetheless!

**magpi.cc/2hg1KpT**

This should look familiar – it's our cover from last issue! It was created by Lenard Gunda (**twitter.com/lenardg**) by following our tutorial. Did you also make our cover light up?

# CROWDFUND THIS!

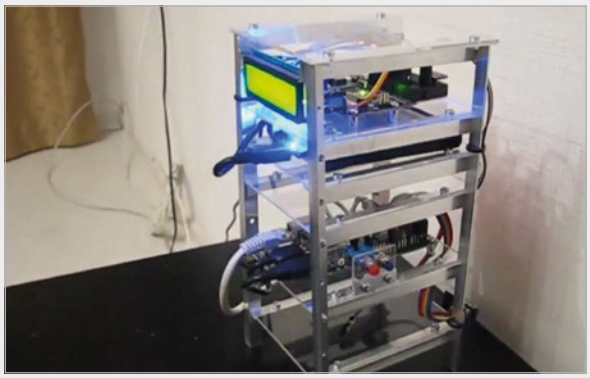
The best crowdfunding hits this month for you to check out...



## GRASPI

**kck.st/2fWNAs2**

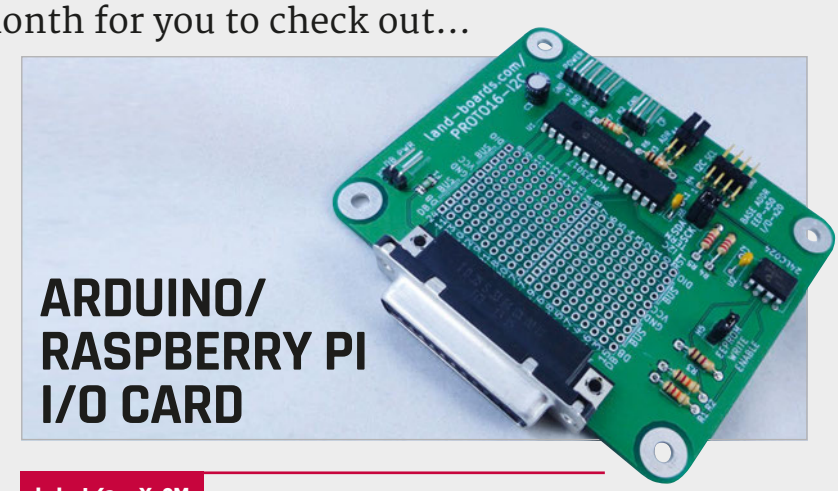
Sonic Pi is great, but so are guitars. GRasPi is a very interesting concept, taking a portable, battery-powered 15W guitar amplifier and making it work with a Raspberry Pi. This way, you can add effects and tie them to a pedal. It all runs on open-source software as well, which is pretty great. The campaign is asking for a decent amount of money, though (these things aren't cheap to build!), so have a look to see if it's for you.



## 9RACK

**magpi.cc/2gBsFsk**

The pitch for this campaign is that conventional racks are large and inconvenient, so why not create a more DIY approach with something a little smaller that you can use at home if you want? Also, it's built around the size of a Raspberry Pi. As the name suggests, you can easily store nine Raspberry Pis on the rack when built, and its simple design means you get pretty good access to all sides of the Raspberry Pi.




## ARDUINO/ RASPBERRY PI I/O CARD

**kck.st/2ggXe8M**

A prototyping board that connects to any microcontroller or microcomputer with an I²C interface – which the Raspberry Pi has! It's a very well-thought-out piece of kit which greatly extends the I²C of the Raspberry Pi, with loads of connectors and a 16-bit processor on board. It's already been funded at the time of writing, so if you want to jump on board and give this board a go, now's the time!

# BEST OF THE REST

Here are some other great things we saw this month

**magpi.cc/2gBx2na**

### THWOMP CASE

A retro console built out of cement? Geeksmithing thought it was a good idea, and went the extra mile to make it look like the Thwomp hazard from *Super Mario*. We advise you not to put it above your toilet in case a plumber comes round to fix it.

**magpi.cc/2gBu6a1**

### CHICKEN POT PI

We giggled a bit over this when it entered our feeds. What do you do when you desperately want a case for your Raspberry Pi? Apparently stick it in a stock cube box, not the box the Pi came in. Just saying.

# COMMUNITY PROFILE

# JILLIAN OGLE

The mother of adventure-ready robots, Pokémon catchers, and helmets

## Jillian Ogle

**Category:** Founder and CEO of Let's Robot

**Day job:** Creator

**Website:** letsrobot.tv, magpi.cc/2gEZp3S

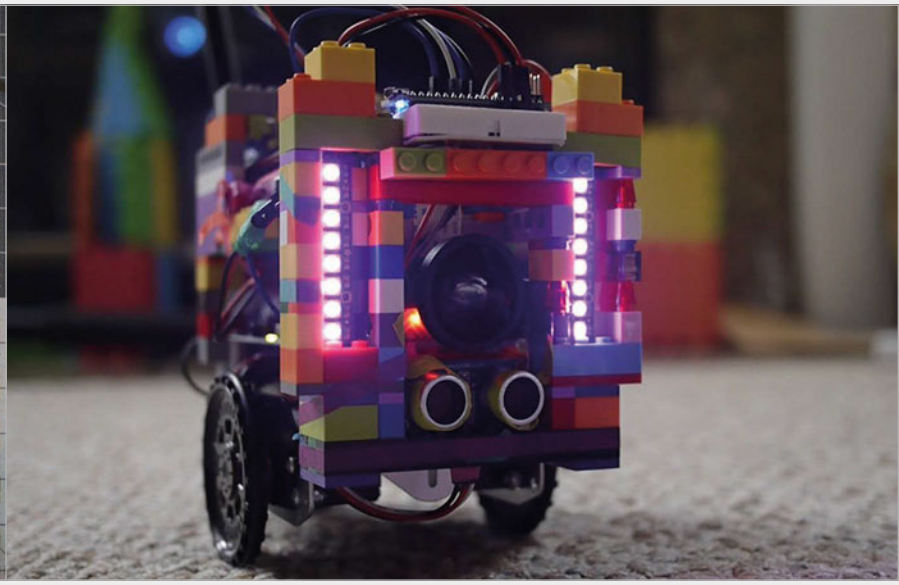
Let's Robot streams twice a week, Tuesdays and Thursdays, and allows the general public to control a team of robots within an interactive set, often consisting of mazes, clues, challenges, and even the occasional foe. Users work together via the Twitch.tv platform, sending instructions to the robots in order to navigate their terrain and complete the set objectives. Aylobot, the first robot of the project, boasts a LEGO body, while Ninabot, the somewhat 2.0 upgrade of the two, has a gripper, allowing more interaction from users. Both robots have their own cameras that stream to Twitch, so that those in control can see what they're up to on a more personal level; several new editions have joined the robot team since then, each with their own unique skill.

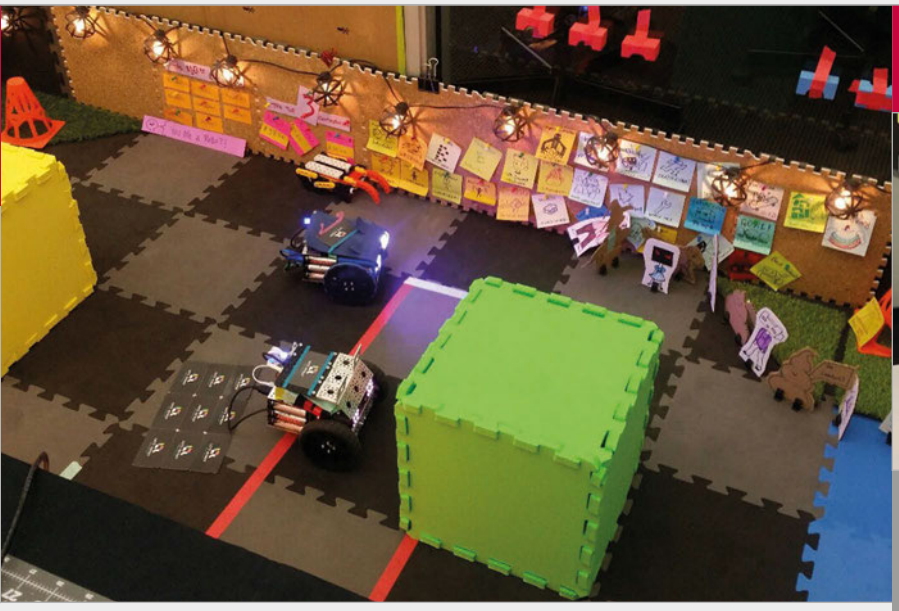
Let's Robot is the brainchild of Jillian Ogle, who originally set out to make "the world's first interactive live show using telepresence robots collaboratively controlled by the audience". However, Jill discovered quite quickly that the robots needed to complete the project simply didn't exist to the standard required… and so Let's Robot was born.

After researching various components for the task, Jill decided upon the Raspberry Pi, and it's this small SBC that now exists within the bodies of Aylobot, Ninabot, and the rest of the Let's Robot family.

**Below** "Post-Its I drew for our #LetsRobot subscribers. We put these in the physical sets made for the robots. I still have a lot more to draw…"

**Below Right** Let's Robot aims to change the way we interact with television, putting the viewer in the driving seat





In her previous life, Jill worked in art and game design, including a role as art director for Playdom, a subsidiary of Disney Interactive; she moved on to found Aylo Games in 2013 and Let's Robot in 2015. The hardware side of the builds has been somewhat of a recently discovered skill, with Jill admitting, "Anything I know about hardware I've picked up in the last two years while developing this project."

More recently, as Let's Robot continues to grow, Jill can be found sharing the antics of the robots across social media, documenting their quests such as the hilarious attempt to create party invites and the more recent Hillarybot vs Trumpbot balloon-head battle, where robots with extendable pin-mounted arms fight to pop each other's head.

**Above** The interactive sets take up a large part of the space at The Batchery, a 'Global Startup Incubator' in Oakland, California.

Alongside the robots, Jill has created several other projects that both add to the interactive experience of Let's Robot and comment on other elements of social trends out in the world. Most notably, there is the Pokémon Go Robot, originally a robot arm that would simulate the throw of an on-screen Poké Ball. It later grew wheels and took to the outside world, hunting down its pocket monster prey. It's also worth noting other builds, such as the WiFi Livestream Goggles that Jill can be seen sporting across several social media posts. The goggles, with a Pi camera fitted between the wearer's eyes, allow viewers to witness Jill's work from her perspective. It's a great build, especially given how open the Let's Robot team are about their continued work and progression.

> "After researching various components for the task, Jill decided upon the Raspberry Pi"

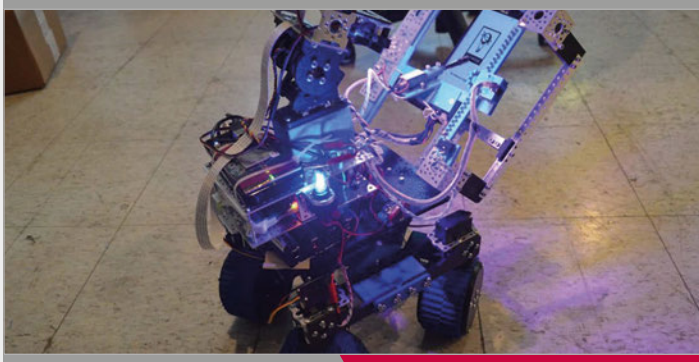
And finally, one project we are eager to see completed is the 'in production' Pi-powered transparent HUD. By incorporating refractive acrylic, Jill aims to create a see-through display that allows her to read user comments via the Twitch live-stream chat, without having to turn her eyes to a separate monitor.

# HIGHLIGHTS

### LET'S ROBOT

twitch.tv/letsrobot

Twice a week, the robots are controlled by the viewers, allowing them the chance to complete tasks such as force-feeding the intern, attempting to write party invitations, and battling in boss fights.



### POKÉMON GO ROBOT

magpi.cc/2hg0ovz

Originally sat on a desk, the Pokémon Go Robot earned itself a new upgrade, gaining the body of a rover to allow it to handle the terrain of the outside world. Paired with the Livestream Goggles, viewers can join in the fun.

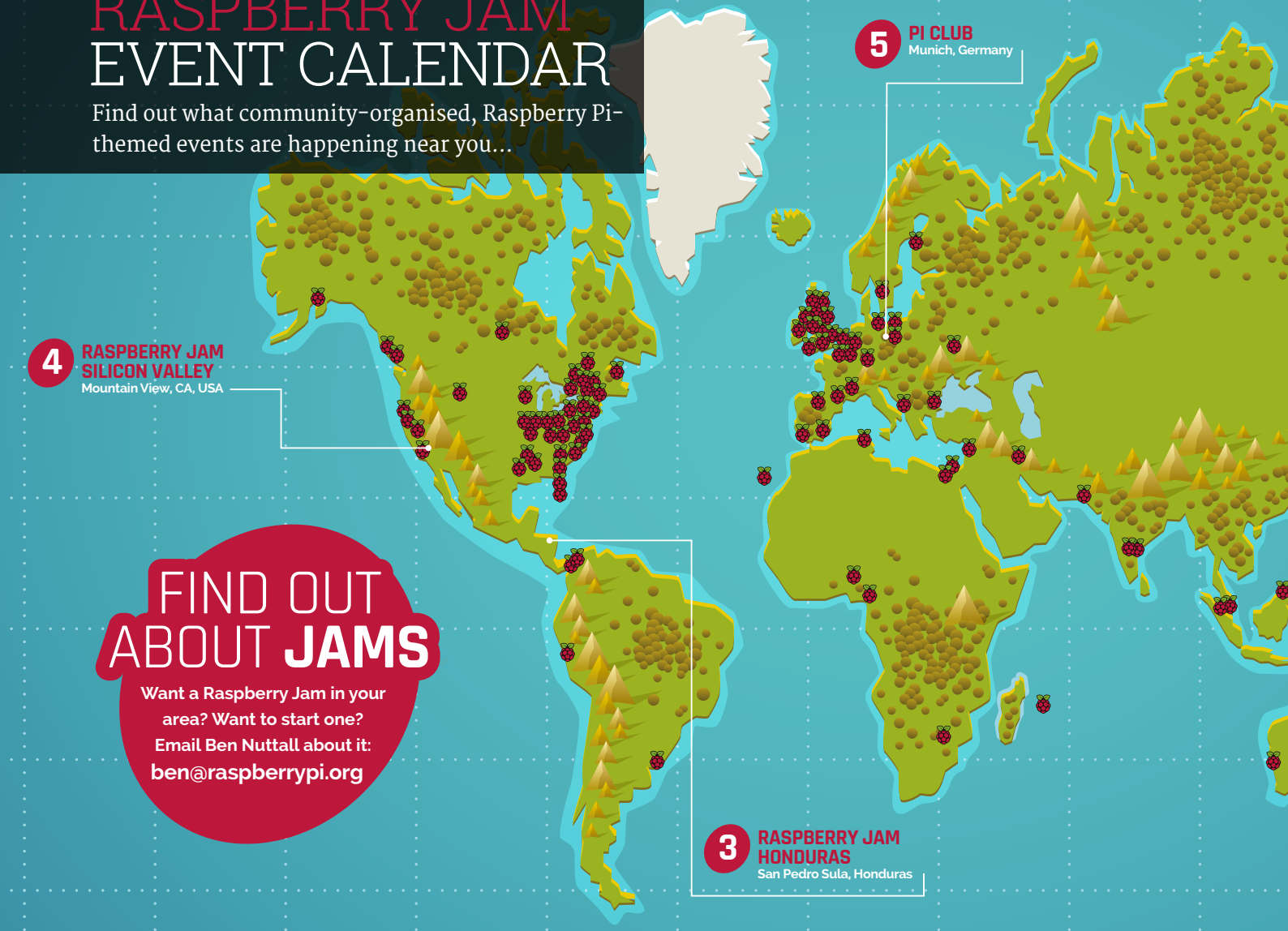
### THE LIVESTREAM GOGGLES

magpi.cc/2hfWe6T

The WiFi-enabled helmet allows viewers the ability to see what Jill sees, offering a new perspective alongside the Let's Robot bots. The Raspberry Pi camera fits perfectly between the eyes, bringing a true eye level to the viewer. She also created internet-controlled LED eyebrows... see the video!

# RASPBERRY JAM EVENT CALENDAR

Find out what community-organised, Raspberry Pi-themed events are happening near you...




## FIND OUT ABOUT **JAMS**

Want a Raspberry Jam in your area? Want to start one? Email Ben Nuttall about it:
ben@raspberrypi.org

## HIGHLIGHTED EVENTS

### 1 MANCHESTER RASPBERRY JAM

**When:** Saturday 14 January
**Where:** The Shed, Manchester, UK
**magpi.cc/2fW9ag8**
Learn how to get started with the Raspberry Pi before attending workshops to learn more.

### 2 MALVERN RASPBERRY PI JAM

**When:** Wednesday 18 January
**Where:** Wyche Innovation Centre, Malvern, UK
**magpi.cc/2fWb40e**
Show off your Raspberry Pi projects or just come to learn more about the Pi.

### 3 RASPBERRY JAM HONDURAS

**When:** Friday 20 January
**Where:** San Pedro Sula, Honduras
**magpi.cc/2fWkLf5**
An event for students to share knowledge of programming and electronics, and learn new things.

### 4 RASPBERRY JAM SILICON VALLEY

**When:** Saturday 21 January
**Where:** Computer History Museum, Mountain View, CA, USA
**magpi.cc/2fW5KKg**
Find out about the Raspberry Pi and see other microcomputers at the Silicon Valley Jam.

## REGULAR EVENTS

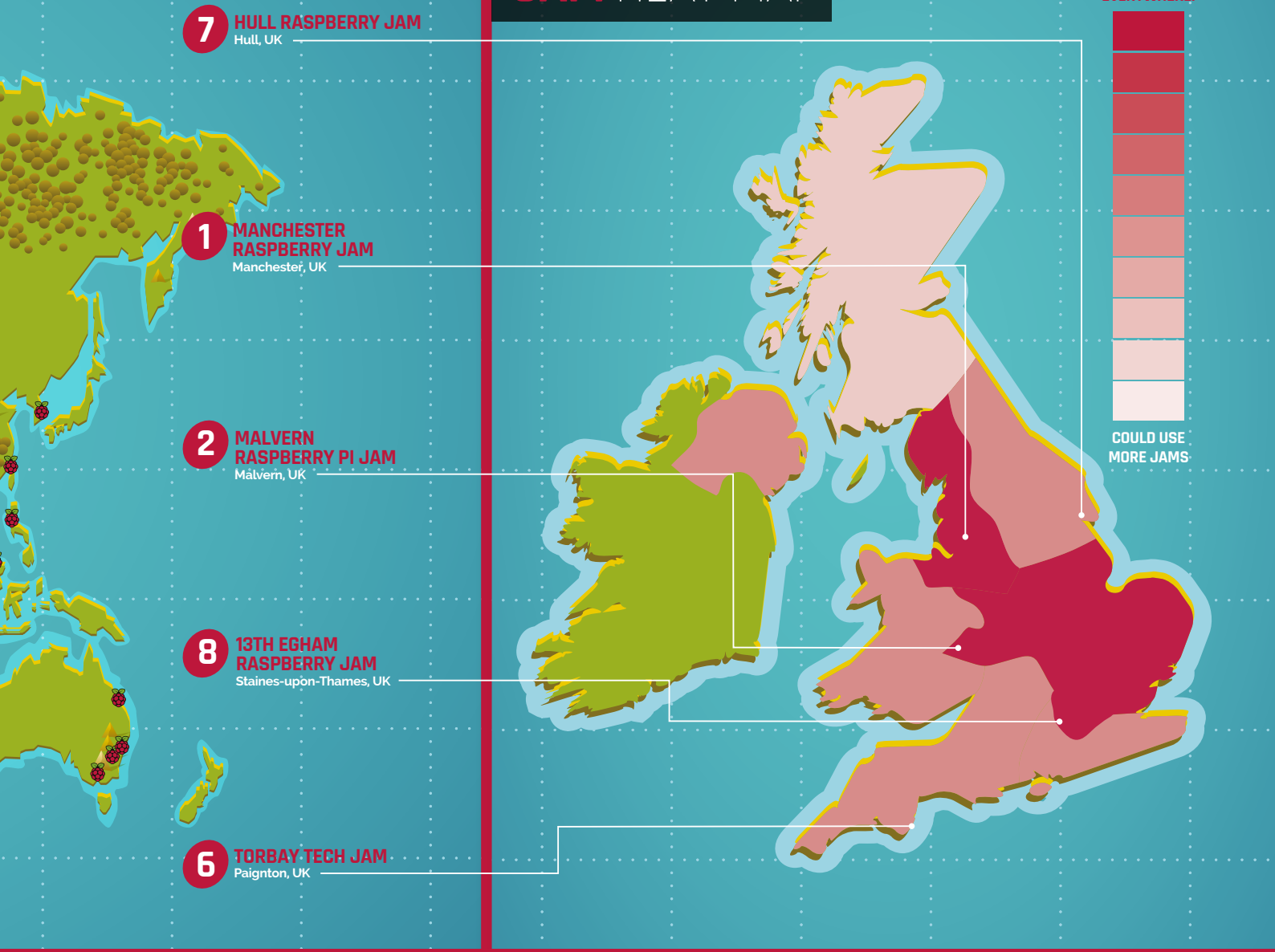
### 5 PI CLUB

**When:** Wednesday 4 January
**Where:** Fab lab inventors, Munich, Germany
**magpi.cc/2fW3l2q**
The last of a series of regular Jams that have been held every Wednesday and Friday.

### 6 TORBAY TECH JAM

**When:** Saturday 14 January
**Where:** Paignton Library and Information Centre, Paignton, UK
**torbaytechjam.org.uk**
This fun, family-friendly event aims to inspire people of all ages to get into coding and take up STEM subjects.

# JAM HEAT MAP




## 7 HULL RASPBERRY JAM

**When:** Saturday 21 January
**Where:** Malet Lambert School, Hull, UK
**magpi.cc/2fW2Epy**
Get hands-on with digital making activities through workshops and a hackspace area.

## 8 13TH EGHAM RASPBERRY JAM

**When:** Sunday 22 December
**Where:** Gartner UK HQ, Staines-upon-Thames, UK
**magpi.cc/2fW3CSJ**
Post-Christmas, people will have new Raspberry Pis and components for making exciting projects.

# FILL IN THE GAPS!

## SOUTH WEST NEEDS RASPBERRY JAMS

While there are plenty of Raspberry Jams in the South East and Midlands, it's a bit harder to track down Jams in the farthest reaches of the South West of England. The regular Torbay Tech Jam is excellent, but it would be great to have a few more Jams going on! Find out more info on setting up Jams from previous issues and by heading to **raspberrypi.org/jam**.

# YOUR LETTERS

## Great Pi North

**I'm a retired person in Western Canada who wants to help young people to get excited about coding and working with Raspberry Pi. There aren't many Raspberry Jams here in Canada, though, especially where I live. I have a few questions about how to go about setting up my own event.**

**Firstly, I need to get hold of your most elementary of elementary books on starting out. These will help me to get ready and also to be able to teach the kids some stuff with the Raspberry Pi.**

**Secondly, if I have a Saturday workshop for youth, do you have someone who could participate via video call? It might help drive attendance and then I can get someone who really knows what they're doing.**

**James Bennett**

It's great you're looking to run a Jam or gathering of some kind! North American Jams aren't as common as they are here in the UK, so it would be a great addition to the world map of events. Before we answer your questions, it might be worth reading back a few issues and looking at the Raspberry Jam interviews we did. You might pick up a few extra tips!

As for books, there are a few you can get your hands on. We have produced plenty, such as the range of Essentials books (including how to use the command line, programming a game in Python, using Scratch, and more), and our official Projects Books always come with help on getting started for people new to Raspberry Pi. You can find all the information on them on our website under Catalogue.

In terms of getting someone to call in, that might be very tricky. The teams are not that big and are usually very busy so are unlikely to have the time to do it.

Hopefully we've pointed you in the right direction for starting up your own event, though!

If you ever want to find out more info on Jams, there are great links on the Events pages

## Download all the code

**Is there any way you could provide a ZIP file of all the code in each issue on your website when it's released? This would make it a lot easier to obtain the code when you want to use it or copy it.**

**Leonard Sparrow**

Right now we like to make sure all the code listings for the tutorials in the magazine exist somewhere online, usually via GitHub, so in the short term and for previous issues it might be best to follow those links to get the code and such.

However, that is a pretty good idea. We have been thinking about putting all the code up on GitHub for each issue, and providing it all as an extra ZIP file on our GitHub or directly from our website shouldn't be too much of an issue. Take a look on the page for this issue on our website and we'll trial putting all the code into a ZIP file for everyone.

## More to do with C

**With the release of the special issue *Learn to Code with C*, I was quite happy. I am learning how to program in C with the intention of using it with microcontrollers and eventually Java with Android.**

**What I was hoping to see in your release was some information on RPi GPIO with C. I did a little searching and found that there could easily have been one whole special issue on this topic. Here's hoping to see that in the near future.**

**One thing that I noticed is that there are some differences with the way that the Raspberry Pi reacts to printf formatting for pointers. Using the same program as found in the Pointers chapter, on Ubuntu MATE, gcc provides a warning, requires long integer formatting to gives the correct values, and actually needs pointer formatting to provide no compiler warnings at all. I gather the Raspberry Pi is set up in a way that is unique in some aspects.**

**I realise that the onus is on the programmer to find all this stuff out, but it might be worth a mention somewhere along the way. The other thing I was hoping to find out is that C is as easy to use as Python with the Raspberry Pi. I have seen the WiringPi site, as well as a tutorial on SparkFun.**

**Bruce Fleming**

We're glad you like the C Essentials book: it seems like it has interested a lot of people and got them coding. We're definitely not done with C either and getting it working with the Pi's GPIO is definitely something we'll be considering in the future, so keep your eyes peeled.

As for differences between systems, this is a tricky one as there's not much you can do about it. There can even be subtle differences between IDEs. We're sure all the code works as expected on Raspbian, though, as that's the main OS for all our Raspberry Pi stuff.

Finally, Python is generally considered rather more straightforward to learn than C, which is why a lot of Pi stuff uses it!

# FROM THE FORUM: TEMPORAL ANOMALY

The Raspberry Pi Forum is a hotbed of conversations and problem-solving for the community – join in via **raspberrypi.org/forums**

**I got my *MagPi* issue 51 today here in California, USA and there's a lot of good content; I enjoyed the 'build a robot' article. I see it promises 'Spooky Halloween Projects' on the cover, which seems an odd choice given the delivery date (1 November). Only 364 more days to go!**
**jbeale**

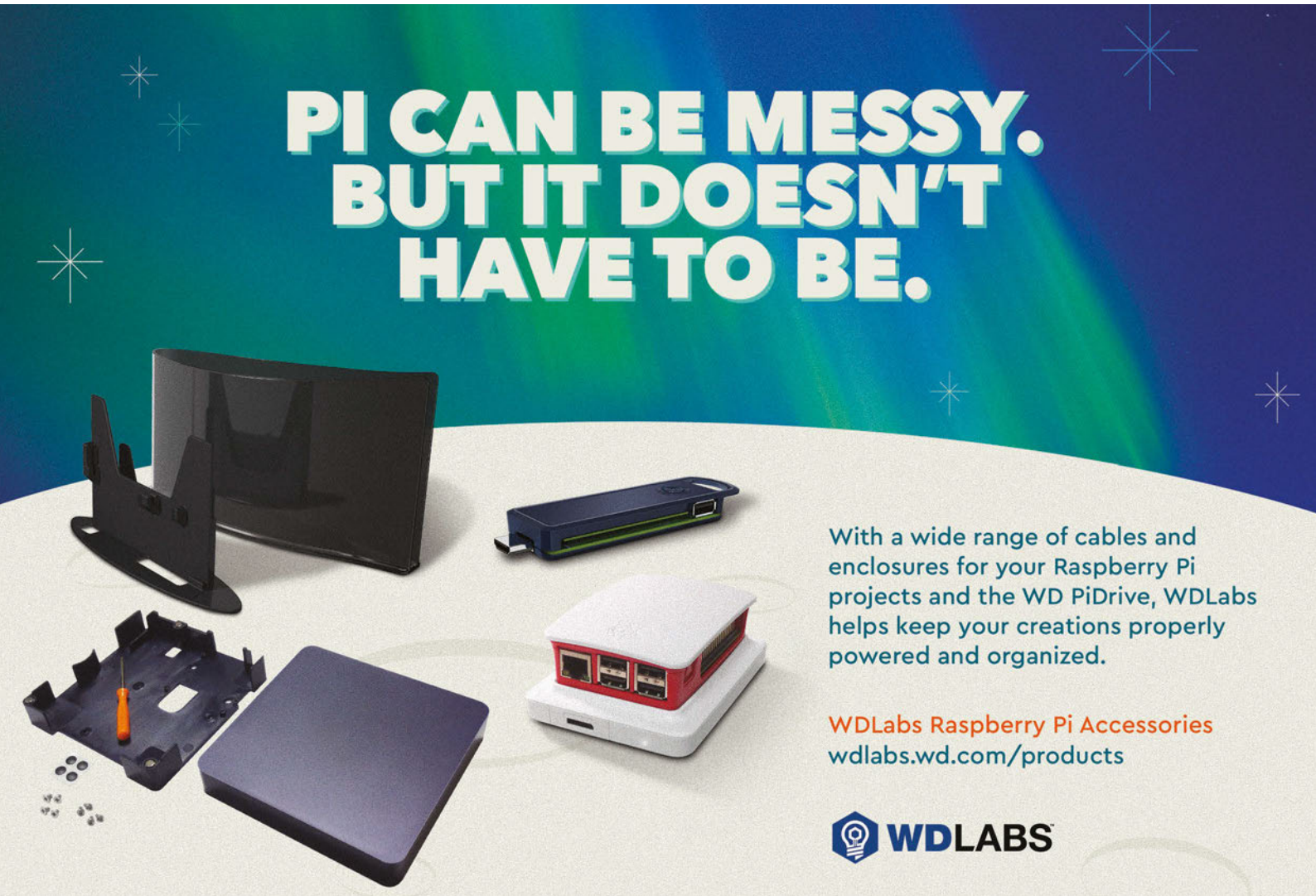
Ah, yes: unfortunately the delay in getting around the world can make some of the more extremely time-sensitive topics a bit harder to get right. In the UK the mag did come out only the weekend before Halloween.

Luckily, as you may have noticed from last issue, the Christmas content came out in plenty of time for the holiday season. Usually we try to make sure content which is seasonally themed in this manner comes out well in advance – for example, last year our Halloween feature was in the October issue, which came out right at the end of September.

Hopefully this explains why it happened and that it shouldn't occur too often. At least you now have a whole year to plan out your next Halloween projects!

## WRITE TO US

**Have you got something you'd like to say?**
Get in touch via **magpi@raspberrypi.org** or on The MagPi section of the forum at: **raspberrypi.org/forums**


PI CAN BE MESSY.
BUT IT DOESN'T
HAVE TO BE.
With a wide range of cables and enclosures for your Raspberry Pi projects and the WD PiDrive, WDLabs helps keep your creations properly powered and organized.
WDLabs Raspberry Pi Accessories
wdlabs.wd.com/products
WDLABS

# WIN

The MagPi

# A MYSTERY MAKER BOX FULL OF GOODIES!

We've had a great year full of excellent Raspberry Pi kits and add-ons and other bits, a treasure trove for makers of all skill levels. We've put some of them into a mystery box that one lucky reader will be able to win. What's inside the box? You'll have to win it to find out...

## WHICH PROJECT CAME FIRST IN ISSUE 50'S GREATEST RASPBERRY PI PROJECTS ARTICLE?

**Tell us by 23 January for your chance to win!**

Simply email **competition@raspberrypi.org** with your name, address, and answer!

### Terms & Conditions

Competition closes on 23 January 2017. Prize is offered worldwide to participants aged 18 or over, except employees of the Raspberry Pi Foundation, the prize supplier, their families or friends. Winners will be notified by email after the draw date. By entering the competition, the winner consents to any publicity generated from the competition, in print and online. Participants agree to receive occasional newsletters from The MagPi magazine (unless otherwise stated upon entry). We don't like spam. Participants' details will remain strictly confidential and won't be shared with third parties. Prizes are non-negotiable and no cash alternative will be offered.

# Love your Pi? Love Music?

Designed by audio experts, enjoyed by everyone

Made in the UK

10% Discount voucher "MagPi10"

**Jaw dropping audio quality for your Raspberry Pi**

Connect Headphones, Speakers, RCA, Toslink, S/PDIF or XLR

We work with all the major music solutions to ensure compatibility

Sonic Pi · moOde audio player · OSMC · piCorePlayer SQUEEZE AUDIO · VOLUMIO · roon · Archphile · PiMusicBox · KODI · rune audio · M2P MAX2PLAY

Twitter: @IQ_audio
Email: info@iqaudio.com
Web: www.iqaudio.com

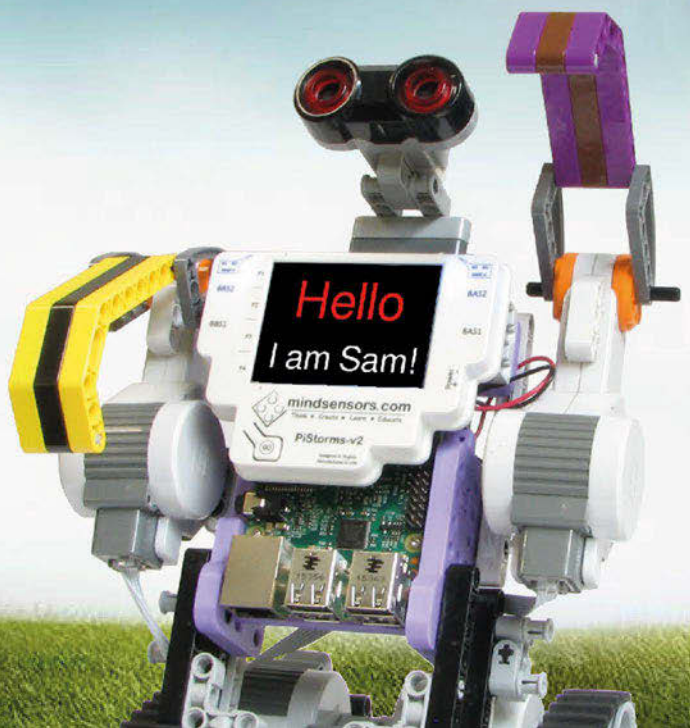
IQaudIO

IQaudio Limited,
Cricklade Wiltshire.
Company No.: 9461908

mindsensors.com

Think • Create • Learn • Educate

!! Special !!
15% discount:
Use Code:
MAGPISPECIAL




PiStorms-v2

Make Stunning Robots with LEGOs and Raspberry Pi !

**MATT RICHARDSON**

Matt is Raspberry Pi's US-based product evangelist. Before that, he was co-author of *Getting Started with Raspberry Pi* and a contributing editor at *Make:* magazine.

# MAKING RESOLUTIONS

**Matt Richardson** discusses how to rededicate yourself to digital making in 2017

I floss my teeth every day. It's the only New Year's resolution that I've ever managed to keep, and I've now kept that promise to myself for many years. Admittedly I've made plenty of other resolutions, but they haven't managed to stick. Even so, with 2017 around the corner, I thought it would be fun to discuss some of the possible resolutions for all of us as digital makers.

Of course, so many resolutions are a recommitment to healthier living, and there's no shortage of ways that digital making with Raspberry Pi can help us become healthier. In *The MagPi* issue 41, we featured Dot Silverman's Motivational Bathroom Scale project. It's a Raspberry Pi with a speaker hooked up to a bathroom scale. If you're not meeting your weight-loss goals, it'll berate you with sassy comments to motivate you to get back to the gym. You can, naturally, reprogram it to use positive reinforcement instead if you prefer.

You could also use a Raspberry Pi Zero with an accelerometer to create your own pedometer, or hack your stationary bike so that you can only turn on the TV every day after you've cycled for 25 minutes. There are tons of ways that Raspberry Pi might be able to help you to live a healthier lifestyle.

Along the lines of self-improvement, many people's resolutions relate to learning. As digital making and learning go hand-in-hand, there's an opportunity here to rededicate ourselves to learning more with Raspberry Pi. Try out a new programming language or Python library. Experiment with an electronic component you've never used. Learn to solder if you don't know how to already. Dedicating weekly time to gaining a deeper understanding of some aspect of technology is something I would like to do in 2017.

Even better than resolving to teach yourself something new, you can also make a commitment to teaching others. You could volunteer at your local Code Club, teach a workshop at a Raspberry Jam, or help your local school. You could even sit down with a friend or family member and show them a thing or two about Linux and programming. I would encourage you to use all the free educational resources on our site to help you do that.

## Be inspired

Are there any projects on your shelf that remain unfinished? Perhaps it's time to take stock of those and commit to finishing them this year before taking on any new projects. Or if you're looking for new project ideas, I've found that getting into the habit of daily brainstorming helps a great deal. You can keep a notebook and write out a page of ideas every day. It doesn't matter if they're practical or absolutely crazy. Just put pen to paper and get yourself thinking about what you want to make. I've found that when you get into the habit of challenging yourself to think creatively on a regular basis, the ideas start to come much more easily.

There are many possible ways in which digital making and New Year's resolutions coalesce. Even if you're like me and you've broken more resolutions than you've kept, don't let that discourage you. It's certainly better to take stock of things and at least try to rededicate yourself to improvement as opposed to giving up before you've even started. In other words, a broken New Year's resolution is better than one not made at all. Even if you end up sticking to just one resolution out of 30, you're probably better off. I think at least my dentist would agree.

# The MagPi ESSENTIALS

LEARN | CODE | MAKE

# OUT NOW IN PRINT

ONLY **£4/$6**

*from*

**raspberrypi.org/magpi**

The MagPi ESSENTIALS From the makers of the official Raspberry Pi magazine

GET THEM DIGITALLY:

Available on the App Store

GET IT ON Google Play

# THE NEW SB COMPONENTS ROBOT CHASSIS RANGE

SB Components

CREATE YOUR OWN
**REMOTE CONTROLLED CAR**
OR **OBSTACLE-SENSING ROBOT**

**The perfect companion** for the SB Components' **Motor-Shield**

**FEATURES INCLUDE:**

- Light-weight aluminium
- Electric motors
- Large wheels
- Stackable chassis plates

www.**robots.sb-components**.co.uk Call: **0203 514 0914**

At **SB Components** we strive to offer our customers the best prices for the best products. Our product team works tirelessly to source top quality affordable components from around the world.

Raspberry Pi is a trademark of the Raspberry Pi Foundation. Raspberry Pi not included.

# WANT TO GET NOTICED?

The MagPi

**ADVERTISE IN THE MAGPI MAGAZINE**

REACH THE RIGHT AUDIENCE FAST

- World's #1 Pi magazine
- Block booking discounts
- Free to download & share
- Live links on iOS & Android

The MagPi is the most exciting mag in tech today, boasting one of the biggest & most engaged audiences in the industry.

YOUR OFFICIAL RASPBERRY PI
The MagPi
BUILD A R CONTROL
MERRY CHRISTMAS MAKES
FREE DEBIAN + PIXEL LIVE BOOTING DVD
The MagPi
BEGINNER'S GUIDE TO CODING
RASPBERRY PI CAMERA MODULE 101
THE ONLY PI MAGAZINE WRITTEN BY THE RASPBERRY PI COMMUNITY

FOR MORE INFO EMAIL **MAGPI@RASPBERRYPI.ORG**

# Expand your Pi

## Stackable expansion boards for the Raspberry Pi

### Serial Pi Plus

RS232 serial communication board. Control your Raspberry Pi over RS232 or connect to external serial accessories.

### Breakout Pi Plus

The Breakout Pi Plus is a useful and versatile prototyping expansion board for the Raspberry Pi

### ADC Differential Pi

8 channel 18 bit analogue to digital converter. I²C address selection allows you to add up to 32 analogue inputs to your Raspberry Pi.

### IO Pi Plus

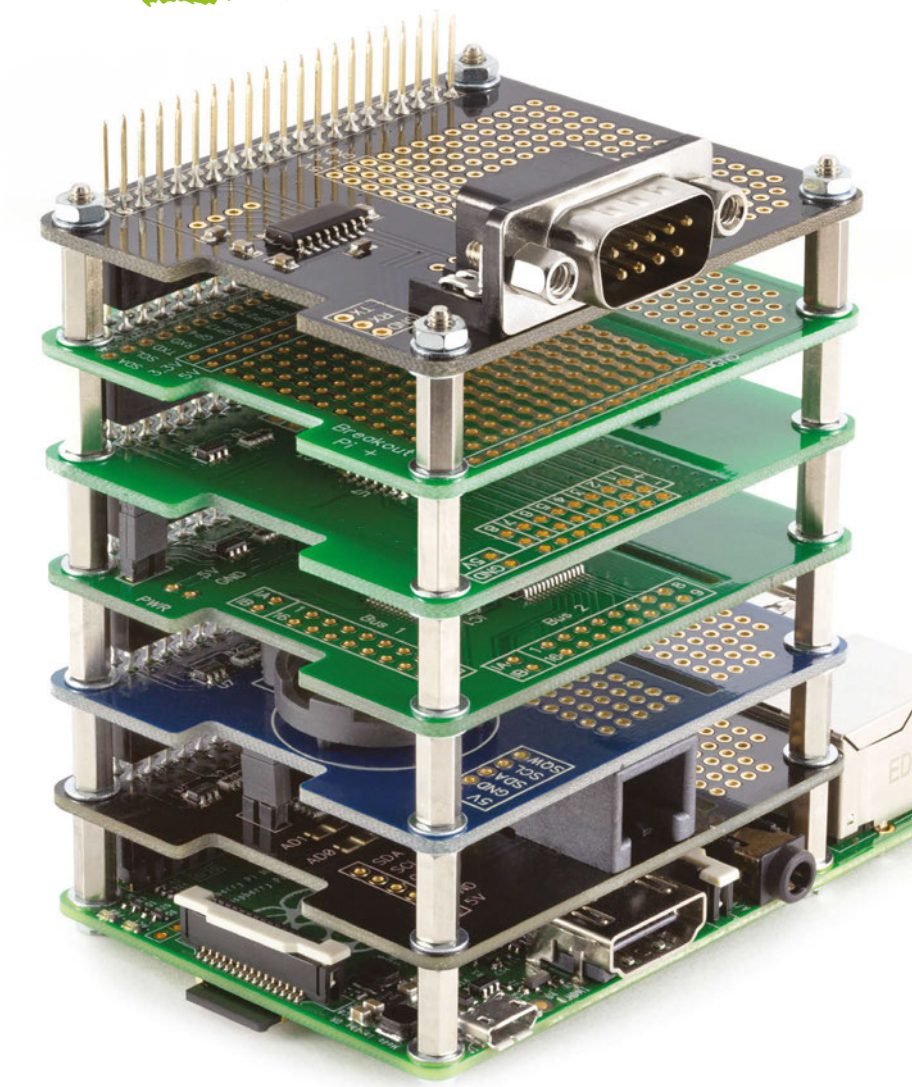
32 digital 5V inputs or outputs. I²C address selection allows you to stack up to 4 IO Pi Plus boards on your Raspberry Pi giving you 128 digital inputs or outputs.

### RTC Pi Plus

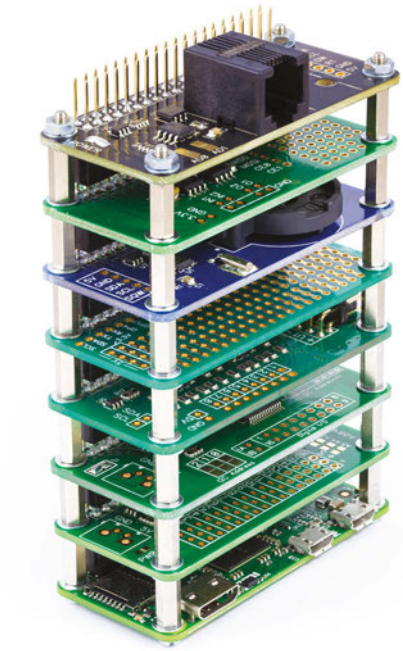
Real-time clock with battery backup and 5V I²C level converter for adding external 5V I²C devices to your Raspberry Pi.

### 1 Wire Pi Plus

1-Wire® to I²C host interface with ESD protection diode and I²C address selection.



Also available for the **Pi Zero**

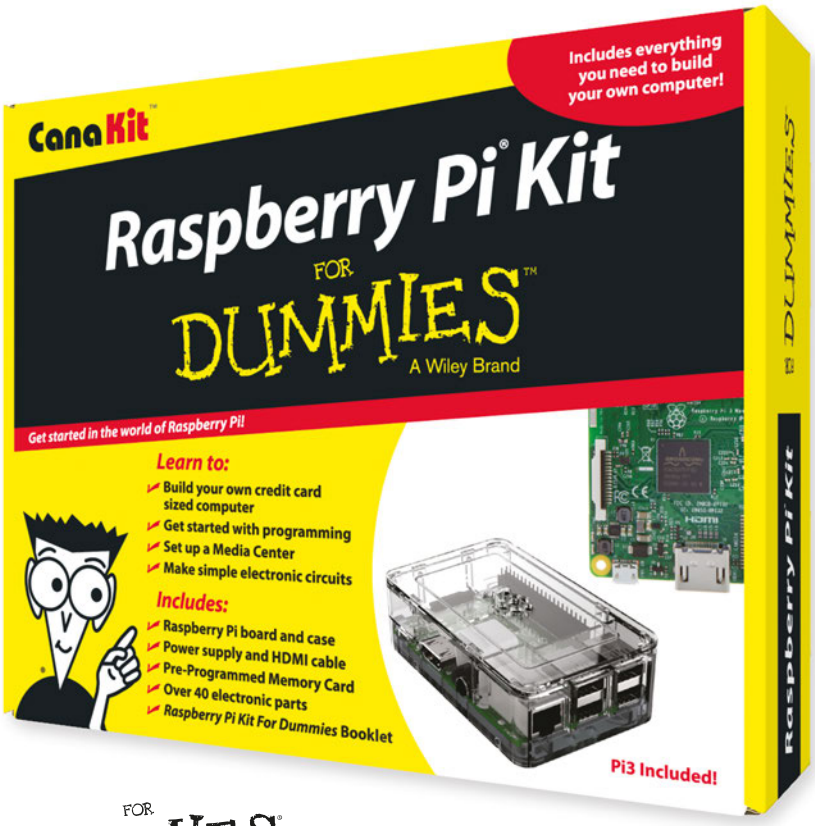


ABelectronics UK

www.abelectronics.co.uk

CanaKit™

# The Perfect Holiday Gift!




FOR DUMMIES® A Wiley Brand

Available for worldwide shipping at:

**WWW.CANAKIT.COM**

Available in Europe through RS Components

RS

## Kit Includes:

- *Raspberry Pi For Dummies* Booklet
- Raspberry Pi 3 Board
- Memory Card
- Plastic Case
- 2.5A Power Supply
- HDMI Cable
- Resistors
- LEDs
- Push Button Switches
- Prototyping Breadboard
- Jumper Wires
- Heat Sinks

**$89.99** US DOLLARS

**£69.99** EXCLUDING VAT

Raspberry Pi is a registered trademark of the Raspberry Pi Foundation. For Dummies and the Dummies Man logo are trademarks or registered trademarks of John Wiley & Sons, Inc. Used under license. RS logo is a registered trademark of RS Components Ltd. CanaKit is a registered trademark of Cana Kit Corporation.

**ELECTRONIC KITS • ELECTRONIC PARTS • RASPBERRY PI • ARDUINO**